Escuela Politécnica Superior de Jaén

Grado en Ingeniería Electrónica Industrial

UNIVERSIDAD DE JAÉN
Nombre del Centro

Trabajo Fin de Grado

LABORATORIOS VIRTUAL Y REMOTO DE UN SISTEMA BARRA-BOLA EN PLATEA

Alumno: Lucena Alonso, Eduardo

Tutor: Prof. D. Elisabet Estevez Estevez
Dpto: Electrónica y Automática Industrial

Febrero, 2024

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Doña ELISABET ESTEVEZ ESTEVEZ , tutora del Proyecto Fin de Carrera titulado: LABORATORIOS VIRTUAL Y REMOTO DE UN SISTEMA BARRA-BOLA EN PLATEA, que presenta EDUARDO LUCENA ALONSO, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, FEBRERO de 2024

El alumno:

LUCENA ALONSO EDUARDO - 26509163F

Firmado digitalmente por LUCENA ALONSO EDUARDO - 26509163F
Fecha: 2024.02.13 18:56:35 +01'00'

Los tutores:

ESTEVEZ ESTEVEZ ELISABET - 72578245C

Firmado digitalmente por ESTEVEZ ESTEVEZ ELISABET - 72578245C
Fecha: 2024.02.13 19:39:25 +01'00'

RESUMEN

En el presente trabajo, se realiza el desarrollo de dos laboratorios online, uno remoto y otro virtual, de un sistema barra-bola para realizar el control de la posición de la bola.

Para el caso del laboratorio remoto, se ha usado el protocolo RIP (Remote Interoperability Protocol) para poder llevar a cabo las comunicaciones entre el servidor y el cliente, cuya interfaz ha sido desarrollada con el software EJsS (Easy Java/JavaScript Simulations). Además, se ha usado una IP Cam para retransmitir en directo la imagen del laboratorio remoto.

Para el caso del laboratorio virtual, únicamente se ha usado el software EJsS, el cual ha permitido emular el comportamiento de la planta real.

Previamente al desarrollo de estos laboratorios, se ha realizado un estudio en el que se abarcan conceptos de modelado de dinámicas, componentes hardware involucrados y el lazo de control necesario. Este estudio será indispensable para el correcto desarrollo de los laboratorios.

Una vez desarrollados los laboratorios online, se ha llevado a cabo su integración en LMS (Learning Management System), para poder presentarlos al usuario (en este caso el alumno) en forma de prácticas docentes para complementar su formación.

ABSTRACT

In this work, the development of two online laboratories has been carried out, one remote and the other one is virtual, for a ball-and-bean system to control the ball's position.

In the case of the remote laboratory, the RIP Protocol (Remote Interoperability Protocol) has been employed to facilitate communication between the server and the client. The client's interface has been developed using the EJsS (Easy Java/JavaScript Simulations) software. Additionally, an IP Cam has been utilized to livestream the image of the remote laboratory in real time.

In the case of the virtual laboratory, only the EJsS software has been used, allowing the emulation of the real plant's behaviour.

Before the development of these laboratories, a study has been carried out, covering concepts of dynamic modelling, hardware components involved in the model and the design of the control loop.

Once the online laboratories were developed, they were integrated into LMS (Learning Management System) to present them to the user (in this case, the student), in form of practices to complement their education.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. Introducción y objetivos.

El uso de laboratorios docentes es fundamental para la realización de trabajos prácticos en todas las áreas de enseñanza. En los estudios universitarios STEM (Science, Technology, Engineering and Mathematics), este uso es mucho más habitual y relevante [1]. Sin embargo, la puesta en marcha y el mantenimiento de laboratorios lleva consigo un coste elevado, además de ser imprescindible el uso de un espacio físico cuando se trata de un laboratorio presencial. Este último condicionante, se ha visto agravado por la situación de pandemia mundial vivida durante los últimos años recientes, que ha imposibilitado el uso de los espacios universitarios por parte de los alumnos. Sin embargo, gracias a ello se ha podido demostrar y ratificar las amplias ventajas de los laboratorios online dentro de la enseñanza [2].

Los laboratorios online pueden considerarse un claro ejemplo del uso eficaz y beneficioso de las Tecnologías de la Información y las Comunicaciones (TIC) en la enseñanza. Actualmente, el elemento TIC más usado y más determinante en la docencia son las plataformas de docencia o LMS (Learning Management System), también conocidas como entornos de aprendizaje virtual o VLE (Virtual Learning Environment).

Los LMS se encuentran plenamente instaurados en las instituciones universitarias, constituyendo el elemento TIC principal de la docencia actual. Su uso es extensivo no solo en la docencia a distancia, ya que constituye un soporte esencial para la docencia presencial actual. Por estos motivos los estudiantes se encuentran totalmente habituados a su uso. Por lo que, es lógico pensar que cuando los alumnos deban trabajar en los laboratorios online, será beneficioso que se presenten integrados con el LMS institucional que dominan y usan a diario.

Este TFG tiene como objetivo final el desarrollo y creación de dos laboratorios online de un sistema barra-bola, uno virtual y otro remoto [3], siendo ambos integrados en el LMS de la Universidad de Jaén (PLATEA). Permitiendo al alumno acceder a una serie de prácticas incrementales integradas en PLATEA, tal y como se puede observar

en la Figura 1-1, pudiendo así complementar la formación recibida en asignaturas como “Automática Industrial” e “Ingeniería de Control”.

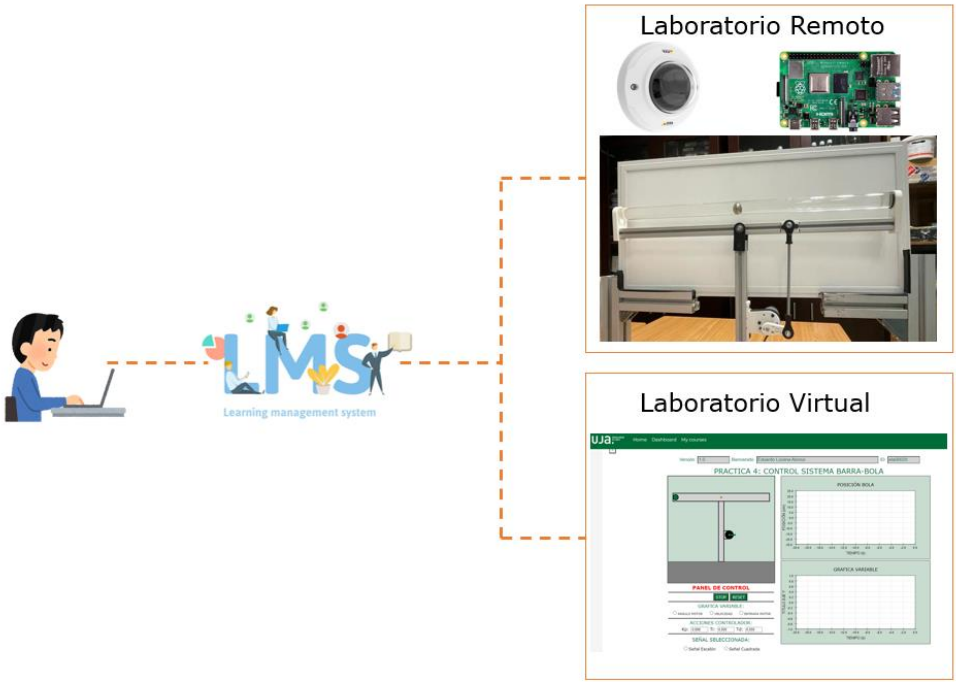

Figura 1-1. Esquema general de laboratorio online

Además, este proyecto, tiene una serie de objetivos específicos, los cuales permitirán alcanzar el objetivo final descrito anteriormente. Dichos objetivos específicos son:

- Modelado: permitirá obtener las dinámicas que conforman el sistema barra-bola, entre las más importantes que se deben obtener se encuentran la dinámica del motor y la dinámica barra-bola.
- Componentes hardware: que forman la maqueta y permitirán el correcto funcionamiento del laboratorio remoto. Además, su correcto entendimiento permitirá la adecuada emulación del laboratorio virtual, debido a los límites físicos de dichos componentes.
- Lazo de control: indispensable para el desarrollo de ambos laboratorios online, permitiendo llevar a cabo el control de la posición de la bola. Además, será necesario seleccionar qué tipo de lazo de control se va a implementar.
- Laboratorios online: consiste en el desarrollo de los laboratorios remoto y virtual basándose en los objetivos específicos anteriores y su posterior integración en LMS.

Este trabajo fin de grado se ha realizado en el mismo orden de la realización del proyecto. En primer lugar, se ha desarrollado el modelado de las distintas dinámicas que forman el sistema barra-bola y que permitirán realizar el control de la posición de la bola.

A continuación, se indicarán todos los componentes electrónicos que forman la maqueta, esto permitirá determinar todos los dispositivos involucrados en el funcionamiento de manera local del sistema barra-bola.

Posteriormente, se llevará a cabo el diseño del lazo de control necesario para poder llevar a cabo el control del sistema.

Seguidamente, se incluirán dos capítulos enfocados al desarrollo de los laboratorios online, uno para el laboratorio remoto y otro para el virtual. En este caso, se ha optado por incluir la integración del laboratorio en LMS dentro del capítulo del laboratorio remoto, ya que, debido a su brevedad, carece de entidad de capítulo.

Por último, se añadirán las conclusiones que han sido extraídas del desarrollo del proyecto y los anexos con la documentación pertinente.

2. Modelado de dinámicas.

En este capítulo se aborda el modelado de las distintas partes que forman el sistema barra-bola, para posteriormente poder llevar a cabo el control de dicho sistema.

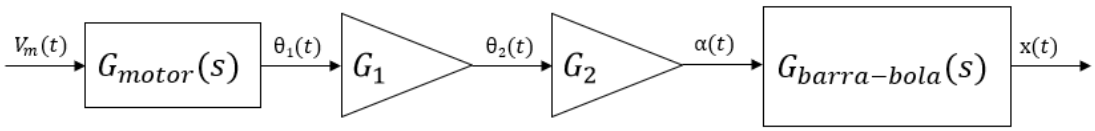

Figura 2-1. Lazo abierto

En la Figura 2-1 se puede observar:

- $G_{motor}(s)$: corresponde a la dinámica del motor, la cual tiene como entrada $V_m(t)$ que corresponde a la tensión de excitación del motor y la salida es la posición del motor en revoluciones $\theta_1(t)$.
- G_1: es una relación lineal, la cual permite obtener la posición del motor en radianes, $\theta_2(t)$, a partir de $\theta_1(t)$.
- G_2: permite obtener el ángulo de inclinación de la barra $\alpha(t)$ a partir de $\theta_2(t)$, al igual que en el caso anterior se trata de una relación lineal.
- $G_{barra-bola}(s)$: corresponde a la dinámica barra-bola, la cual relaciona la posición de la bola x(t), en función del ángulo de inclinación de la barra.

2.1. Modelado del sistema barra-bola.

De esta dinámica se obtiene la relación matemática entre la posición de la bola x(t) y el ángulo de inclinación de la barra $\alpha(t)$.

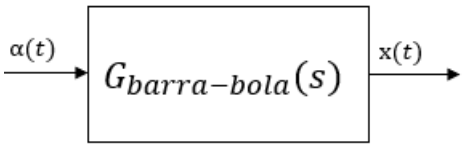

Figura 2-2. Dinámica barra-bola

Antes de proceder con la obtención de la dinámica, es importante aclarar que, debido a la alta velocidad de la dinámica, se ha optado por introducir un líquido de

mayor viscosidad en la barra, este líquido es glicerina rebajada con agua, con el objetivo de ralentizar la respuesta del sistema.

Para obtener esta dinámica, se hará uso de la segunda Ley de Newton [6] y será necesario conocer las distintas fuerzas que existen.

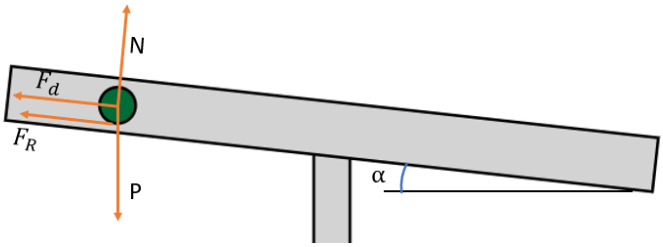

Figura 2-3. Fuerzas presentes en el sistema barra-bola

Tal y como se puede observar en la Figura 2-3, las fuerzas presentes son:

- N(t): fuerza normal, fruto del contacto entre la bola y la barra.
- P(t): peso de la bola.
- F_d(t): fuerza de arrastre del fluido, efecto del líquido de mayor viscosidad introducido en la barra.
- F_R(t): fuerza de rozamiento, producida por la fricción entre la barra y la bola.

A continuación, como ya se ha mencionado, se debe de aplicar la segunda Ley de Newton:

$$\sum \vec{F}(t) = m * \vec{a}(t)$$

Ecuación 1

Para aplicar la Ecuación 1, se ha optado por descomponer las fuerzas en los ejes X e Y.

El primer paso es llevar a cabo la sumatoria de fuerzas en el eje Y. Como se sabe que la bola no tiene desplazamiento en este eje, la aceleración será nula, por lo que, $\sum \vec{F}_y(t) = 0$, de lo que se puede obtener que:

$$N(t) - P_y(t) = 0$$

Ecuación 2

$$P_y(t) = m * g * \cos(\alpha(t))$$

Ecuación 3

$$N(t) - m * g * \cos(\alpha(t)) = 0$$

Ecuación 4

Siendo:

- m: masa de la bola.
- g: constante de gravedad.
- α(t): ángulo de inclinación de la barra respecto a la horizontal.

A continuación, se lleva a cabo la sumatoria de fuerzas en el eje X, pero en este caso sí que hay desplazamiento, por lo que $\sum \vec{F}_x(t) = m * a(t)$. Realizando la sumatoria se obtiene:

$$-F_R(t) - F_d(t) + P_x(t) = m * a(t)$$

Ecuación 5

$$P_x(t) = m * g * sen(\alpha(t))$$

Ecuación 6

$$-F_R(t) - F_d(t) + m * g * sen(\alpha(t)) = m * a(t)$$

Ecuación 7

Lo que se necesita para poder conseguir la dinámica, es expresar la Ecuación 7 únicamente en función del ángulo de inclinación de la barra y de la posición de la bola.

$F_R(t)$, es la fuerza de rozamiento, la cual permite a la bola girar sin deslizar a lo largo de la barra. Este fenómeno recibe el nombre de condición de rodadura [7], el cual relaciona la aceleración con la que se desplaza el centro de masas de un cuerpo y la aceleración angular de rotación de dicho cuerpo. Esto queda reflejado en la Ecuación 8.

$$a(t) = R * a_\alpha(t)$$

Ecuación 8

Siendo:

- a(t), la aceleración lineal de la bola.
- R, el radio de la bola.
- $a_\alpha(t)$, la aceleración angular de la bola.

Aplicando el movimiento de rotación alrededor de un eje que pasa por el centro de masas de la bola:

$$F_R(t) * R = J_{bola} * a_\alpha(t)$$

Ecuación 9

Siendo J_{bola} el momento de inercia de una esfera sólida.

$$J_{bola} = \frac{2}{5} * m * R^2$$

Ecuación 10

Por lo que sustituyendo la Ecuación 10 en la Ecuación 9:

$$F_R(t) * R = \frac{2}{5} * m * R^2 * a_\alpha(t)$$

Ecuación 11

Y sustituyendo la Ecuación 8 en la Ecuación 11:

$$F_R(t) * R = \frac{2}{5} * m * R^2 * \frac{a(t)}{R}$$

Ecuación 12

Si se despeja $F_R(t)$ de la Ecuación 12, se obtiene la expresión:

$$F_R(t) = \frac{2}{5} * m * a(t)$$

Ecuación 13

Sustituyendo la Ecuación 13 en la Ecuación 7, se obtiene:

$$-\frac{2}{5} * m * a(t) - F_d(t) + m * g * sen(\alpha(t)) = m * a(t)$$

Ecuación 14

Tal y como se puede observar en la Ecuación 14, la única expresión que queda por transformar es $F_d(t)$, para ello se aplica la ley de Stokes [8], obteniendo así:

$$F_d(t) = 6 * \eta * \pi * R * v(t)$$

Ecuación 15

Siendo:

- η, la viscosidad del fluido.
- R, radio de giro de la bola.
- v(t), velocidad lineal de la bola.

Para obtener el valor de la viscosidad del fluido, se debe de tener en cuenta que, tal y como ya se ha mencionado previamente, este fluido se trata de una mezcla de agua y glicerina, por lo que:

$$\eta = \eta_{agua} * (p) + \eta_{glicerina} * (1 - p)$$

Ecuación 16

Siendo:

- η_{agua}, viscosidad del agua.
- $\eta_{glicerina}$, viscosidad de la glicerina.
- p, porcentaje de agua en la mezcla. Su valor normalizado es entre 0 y 1, siendo 1 el 100%.

Sustituyendo la Ecuación 15 en la Ecuación 14:

$$-\frac{2}{5} * m * a(t) - 6 * \eta * \pi * R * v(t) + m * g * sen(\alpha(t)) = m * a(t)$$

Ecuación 17

Para simplificar la Ecuación 17, sabiendo que el ángulo de inclinación de la barra va a ser pequeño, se puede tomar $\mathrm{sen}(\alpha(t)) = \alpha(t)$. Por lo que, se obtiene la siguiente expresión:

$$-\frac{2}{5} * m * a(t) - 6 * \eta * \pi * R * v(t) + m * g * \alpha(t) = m * a(t)$$

Ecuación 18

A partir de la Ecuación 18, y sabiendo que la aceleración lineal de la bola se puede expresar como la doble derivada de la posición, y que la velocidad es equivalente a la derivada de la posición:

$$-\frac{2}{5} * m * \frac{d^2x(t)}{dt^2} - 6 * \eta * \pi * R * \frac{dx(t)}{dt} + m * g * \alpha(t) = m * \frac{d^2x(t)}{dt^2}$$

Ecuación 19

Aplicando la transformada de LaPlace a la Ecuación 19:

$$-\frac{2}{5} * m * s^2 * X(s) - 6 * \eta * \pi * R * s * X(s) + m * g * \alpha(s) = m * s^2 * X(s)$$

Ecuación 20

Por último, despejando de la Ecuación 20 la relación entre X(s) y α(s), se obtiene la dinámica del sistema barra-bola que se buscaba:

$$G_{barra-bola}(s) = \frac{X(s)}{\alpha(s)} = \frac{\frac{5}{7} * g}{s * (s + \frac{30}{7 * m} * \eta * \pi * R)}$$

Ecuación 21

Sabiendo que para el caso de estudio del que se dispone, se ha considerado:

- g=9.81 m/s^2
- m=0.05 kg
- R=0.025 m
- Para la viscosidad, se debe de tener en cuenta que el porcentaje de agua es del 70 %, la viscosidad del agua es 0.00105 kg/(m*s) y la de la glicerina es de 1.3923 kg/(m*s). Sustituyendo en la Ecuación 16 se obtiene η=0.418 kg/(m*s).

Se puede obtener la dinámica barra-bola que posteriormente se usará en el desarrollo del laboratorio remoto y digital:

$$G_{barra-bola}(s) = \frac{X(s)}{\alpha(s)} = \frac{7}{s * (s + 2.814)}$$

Ecuación 22

2.2. Modelado del motor.

De esta dinámica se obtiene la relación matemática entre la posición del motor en revoluciones $\theta_1(t)$ y la tensión de excitación del motor $V_m(t)$. Para obtener dicha dinámica, se van a desarrollar dos métodos, haciendo una comparación entre los resultados que ambos ofrecen. Estos métodos son el modelado matemático del motor mediante parámetros experimentales y el modelado matemático del motor a través de su respuesta.

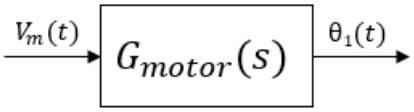

Figura 2-4. Dinámica del motor

2.2.1. Modelado matemático del motor mediante parámetros experimentales.

Para llevar a cabo este modelado [9] lo primero que se debe de obtener son las ecuaciones que rigen el funcionamiento de un motor DC.

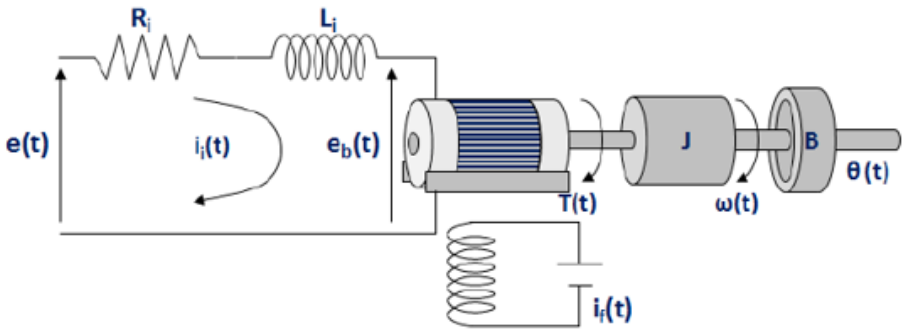

Figura 2-5. Elementos más importantes de un motor DC

En la Figura 2-5, se presentan los elementos más importantes de un motor DC, formando estos un sistema electromecánico.

La parte izquierda de la Figura 2-5 corresponde al circuito eléctrico de la armadura del motor, donde se encuentran la resistencia y la inductancia de la armadura, en estos elementos se producirá una caída de tensión debido a $i_i(t)$. La tensión e(t) corresponde a la tensión con la que se excita al motor, y, por último, se puede observar $e_b(t)$, que es la fuerza contraelectromotriz [voltios]. Esta es una

tensión, producida cuando los conductores de la armadura se mueven a través del flujo de campo establecido del campo $i_f(t)$.

Aclarado esto, y aplicando la Ley de tensiones de Kirchhoff [10] en la malla, se obtiene la siguiente ecuación.

$$e(t) = R_i * i_i(t) + L_i * \frac{di_i(t)}{dt} + e_b(t)$$

Ecuación 23

Despejando de la Ecuación 23 la tensión de la inductancia de la armadura, se obtiene la Ecuación 24.

$$L_i * \frac{di_i(t)}{dt} = e(t) - R_i * i_i(t) - e_b(t)$$

Ecuación 24

Otra ecuación necesaria para el modelado del motor, es la que relaciona la fuerza electromotriz con la velocidad angular del motor a través de la constante contraelectromotriz, k_b [V*s/rad].

$$e_b(t) = k_b * \frac{\partial\theta(t)}{\partial t}$$

Ecuación 25

Siendo θ(t) el ángulo en radianes girado por el motor.

La parte derecha de la Figura 2-5 corresponde al modelo mecánico del motor, siendo J el momento de inercia del motor, B el coeficiente de fricción y T(t) el torque del motor. Aplicando la segunda Ley de Newton, se puede obtener la siguiente ecuación.

$$T(t) = J * \frac{\partial^2\theta(t)}{\partial t^2} + B * \frac{\partial\theta(t)}{\partial t}$$

Ecuación 26

Despejando de la Ecuación 26, la derivada de mayor orden se obtiene la Ecuación 27.

$$J * \frac{\partial^2 \theta(t)}{\partial t^2} = T(t) - B * \frac{\partial \theta(t)}{\partial t}$$

Ecuación 27

La última ecuación necesaria para llevar a cabo el modelado del motor, es aquella que relaciona el torque del motor con la corriente, que circula a través de la malla, a través de la constante de torque, k_p [N*m/A].

$$T(t) = k_p * i_i(t)$$

Ecuación 28

Usando la Ecuación 24, la Ecuación 25, la Ecuación 27 y la Ecuación 28, es posible representar las ecuaciones diferenciales a través de cualquier herramienta, como puede ser Simulink [11].

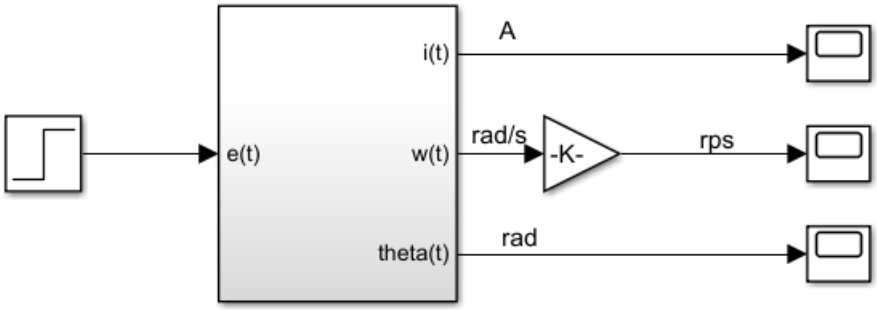

Figura 2-6. Representación simulink

El subsistema de la Figura 2-6 queda reflejado en la Figura 2-7.

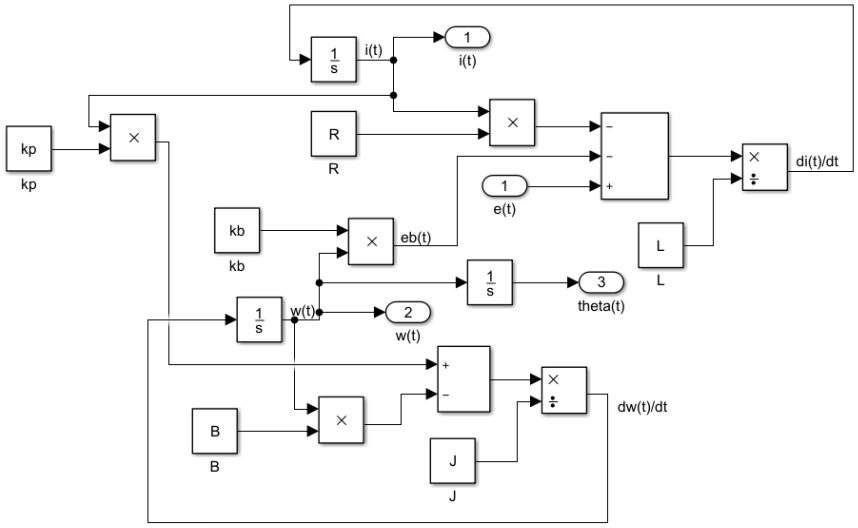

Figura 2-7. Subsistema resultado de las ecuaciones diferenciales

Siendo ω(t) la velocidad angular en rad/s tanto en la Figura 2-6 como en la Figura 2-7.

$$\frac{\partial\theta(t)}{\partial t} = \omega(t)$$

Ecuación 29

Otra manera de representar el sistema es aplicando la transformada de LaPlace [12] a la Ecuación 24, la Ecuación 25, la Ecuación 27 y la Ecuación 28.

$$L_i * s * I_i(s) = E(s) - R_i * I_i(s) - E_b(s)$$

Ecuación 30

$$E_b(t) = k_b * s * \theta(s)$$

Ecuación 31

$$J * s^2 * \theta(s) = T(s) - B * s * \theta(s)$$

Ecuación 32

$$T(s) = k_p * I_i(s)$$

Ecuación 33

Se debe de sustituir la Ecuación 31 en la Ecuación 30 y la Ecuación 33 en la Ecuación 32.

$$L_i * s * I_i(s) = E(s) - R_i * I_i(s) - k_b * s * \theta(s)$$

Ecuación 34

$$J * s^2 * \theta(s) = k_p * I_i(s) - B * s * \theta(s)$$

Ecuación 35

Por último, se debe de despejar $I_i(s)$ de la Ecuación 35 y sustituir en la Ecuación 34, y tras agrupar matemáticamente se obtiene la dinámica que relaciona el ángulo girado del motor con la tensión de excitación del motor.

$$\frac{\theta(s)}{E(s)} = \frac{k_p}{s * [L_i * J * s^2 + (R_i * J + L_i * B) * s + R_i * B + k_p * k_b]}$$

Ecuación 36

Conociendo la Ecuación 29, también se puede obtener la dinámica que relaciona la velocidad angular del motor con la tensión de excitación del motor.

$$\frac{\Omega(s)}{E(s)} = \frac{k_p}{[L_i * J * s^2 + (R_i * J + L_i * B) * s + R_i * B + k_p * k_b]}$$

Ecuación 37

Para confirmar que ambas expresiones ofrecen el mismo resultado tan sólo se debe de llevar a cabo una simulación. Para dicha simulación se ha supuesto que todos los parámetros tienen valor unitario, tal y como se puede observar en la Figura 2-8.

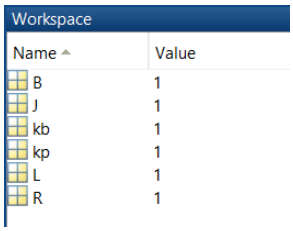

Figura 2-8. Workspace de MATLAB

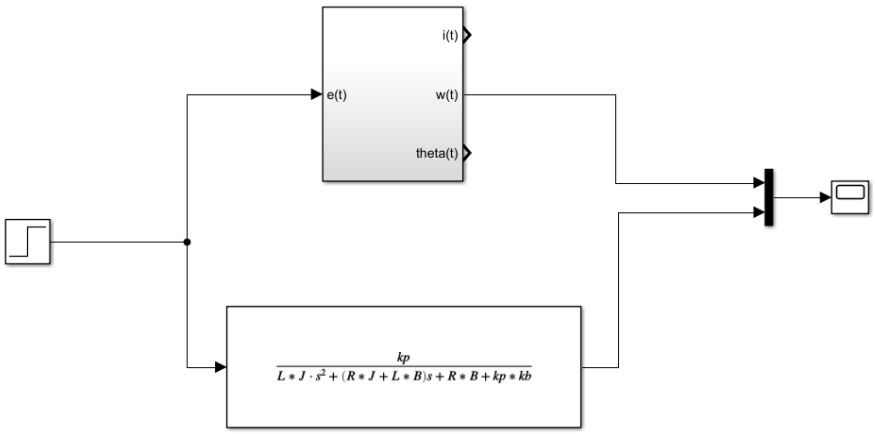

Figura 2-9. Simulación de ecuaciones diferenciales y bloque del motor

Tras llevar a cabo la simulación y visualizando los resultados obtenidos, se puede afirmar que, tal y como era de esperar, ambas respuestas son idénticas, tal y como se puede ver en la Figura 2-10. Se debe de tener en cuenta, que, en dicha respuesta, el valor de la velocidad tiene como unidades rad/s. Por lo que, para obtener la dinámica objetivo, se debería de usar una relación lineal que transforme las unidades a rps.

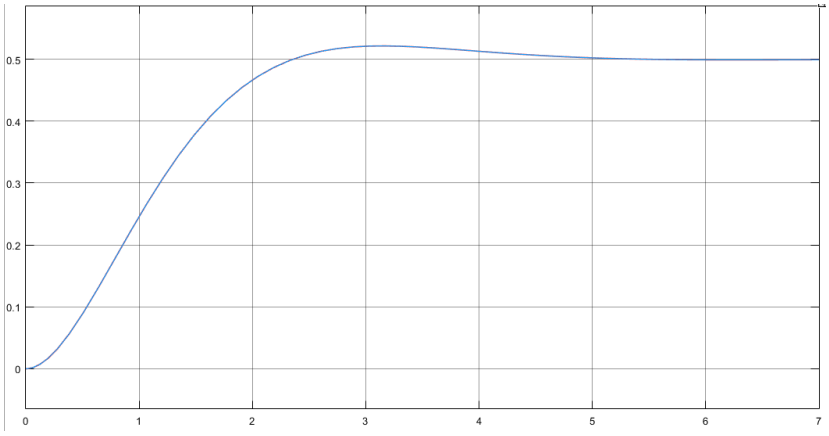

Figura 2-10. Respuestas obtenidas de la simulación de ecuaciones diferenciales y bloque del motor

2.2.2. Obtención de los parámetros experimentales.

Una vez conocidos tanto las ecuaciones diferenciales como el bloque que rigen el comportamiento del motor, se deben obtener dichos parámetros experimentales. Pero para ello primero se debe de conocer el motor del cual se van a obtener los parámetros. Dicho motor es el presente en la Figura 2-11, se trata de un motor de corriente continua, que puede ser alimentado hasta con 12 V, tiene una reductora con una relación 70:1 y un encoder con una resolución de 64 CPR (counts per revolution).

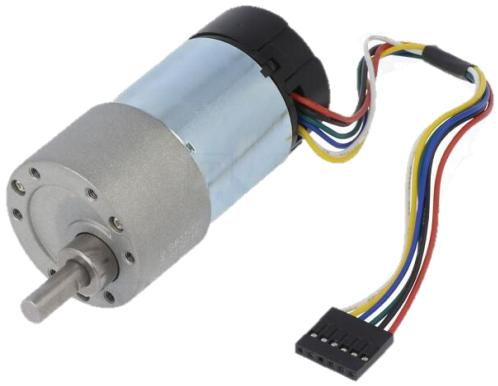

Figura 2-11. Metal Gearmotor 37Dx70L mm 12V with 64 CPR Encoder

El fabricante además facilita la siguiente tabla con las funciones de los distintos cables del encoder.

Color	Function
Red	motor power (connects to one motor terminal)
Black	motor power (connects to the other motor terminal)
Green	encoder GND
Blue	encoder Vcc (3.5 – 20 V)
Yellow	encoder A output
White	encoder B output

Tabla 1. Funciones de los distintos cables del encoder

El conexionado del motor y procesamiento de las salidas del encoder se desarrollará en el capítulo de la Maqueta. Para entender la obtención de los parámetros experimentales con esta información es suficiente.

De esta manera, se pueden obtener los parámetros, comenzando por la resistencia e inductancia de la armadura. Para ello tan sólo se debe de usar un medidor LCR como el que se observa en la Figura 2-12 y medir directamente los devanados de la armadura del motor. Para el caso del motor de la Figura 2-11, habría que medir entre los cables rojo y negro.

Figura 2-12. Medidor LCR

Los siguientes parámetros que se obtendrán son la constante electromotriz y de torque. Se ha de tener en cuenta, que, en unidades del SI, el par motor y las constantes de fuerza electromotriz de retorno son iguales a través de la técnica paramétrico dimensional, cambiando únicamente las unidades de estos parámetros, de esta forma se obtiene la Ecuación 38.

$$k_p = k_b$$

Ecuación 38

De la Ecuación 25 se obtiene la siguiente expresión.

$$k_b = \frac{e_b(t)}{\omega(t)}$$

Ecuación 39

La Figura 2-13 representa la parte eléctrica de la Figura 2-5 ya simplificada, ya que, con alimentación continua, la caída de tensión en la inductancia será nula, por lo que se puede omitir en la representación.

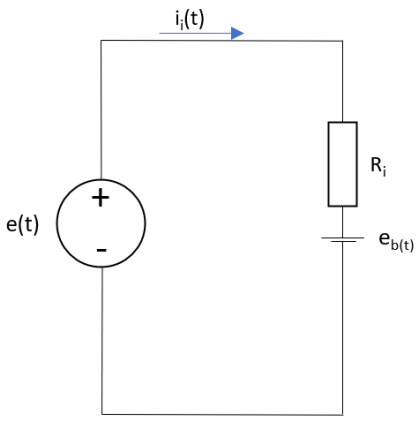

Figura 2-13. Circuito equivalente para obtener constante electromotriz

De esta manera se puede obtener la siguiente expresión.

$$e_b(t) = e(t) - R_i * i_i(t)$$

Ecuación 40

Por lo que sustituyendo la Ecuación 40 en la Ecuación 39 se obtiene la expresión que se necesita para obtener la constante electromotriz.

$$k_b = \frac{e(t) - R_i * i_i(t)}{\omega(t)}$$

Ecuación 41

Para hacer este ensayo, únicamente se necesita una fuente de tensión continua de 12 V, un multímetro y un microcontrolador para procesar la salida del motor. Una vez medidos los parámetros indicados en el montaje de la Figura 2-14, únicamente quedaría sustituir en la Ecuación 41 y obtener la constante electromotriz.

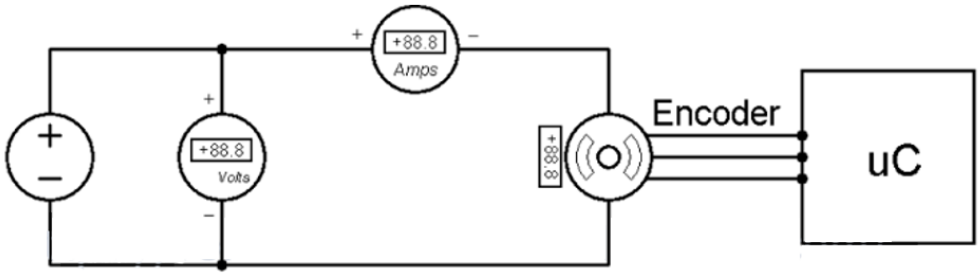

Figura 2-14. Montaje para obtener constante electromotriz

Como se ha comentado anteriormente, la constante electromotriz y la de torque, tienen el mismo módulo, pero se debe de tener en cuenta que las unidades no son iguales.

El siguiente paso será obtener el momento de inercia del motor, para ello se debe de medir el tiempo de estabilización mecánica, t_{mss}. Para obtener este parámetro se debe de usar un osciloscopio, visualizando así el tiempo que tarda la tensión de los devanados del motor en ser invariante en el tiempo.

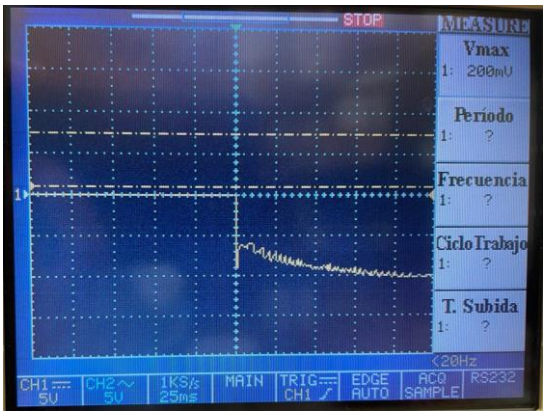

Figura 2-15. Tiempo de estabilización de la tensión

Tal y como se puede observar en la Figura 2-15, existe un régimen transitorio. Una vez obtenido el tiempo de estabilización mecánica, tan sólo queda obtener la constante de tiempo mecánica, la cual se obtiene con la Ecuación 42.

$$t_m = \frac{t_{mss}}{4}$$

Ecuación 42

Con la constante de tiempo mecánica se puede calcular el momento de inercia con la Ecuación 43.

$$J = \frac{t_m * k_p * k_b}{R_i}$$

Ecuación 43

El último parámetro que queda por obtener es el coeficiente de fricción, para ello, usando la Ecuación 27, se hace la segunda derivada nula, ya que se supone régimen estacionario, por lo que no habrá aceleración angular. De esta manera se obtiene la Ecuación 44.

$$B = \frac{T(t)}{\omega(t)}$$

Ecuación 44

Para obtener el coeficiente de fricción, tan sólo se debe de sustituir la Ecuación 28 en la Ecuación 44, obteniendo así:

$$B = \frac{k_p * i_i(t)}{\omega(t)}$$

Ecuación 45

De la Ecuación 45, se conocen todas las variables de los ensayos anteriores, por lo que ya se puede obtener el valor del coeficiente de fricción.

En la se pueden observar los resultados que se han obtenido del motor de la Figura 2-11.

PARÁMETRO	VALOR	UNIDADES
R_i	2.4	Ω
L_i	1.252	mH
k_p	0.6157	N*m/A
k_b	0.6157	V/(rad*s)
J	0.01382	$Kg*m^2$
B	0.01	N*m*s
e(t)	10.5	V

Tabla 2. Resultados obtenidos de los distintos ensayos

Si se lleva a cabo la misma simulación de la Figura 2-9, pero usando los valores de los parámetros medidos y adaptando la salida para que las unidades de la velocidad sean rps, la respuesta que se obtiene queda reflejada en la Figura 2-16.

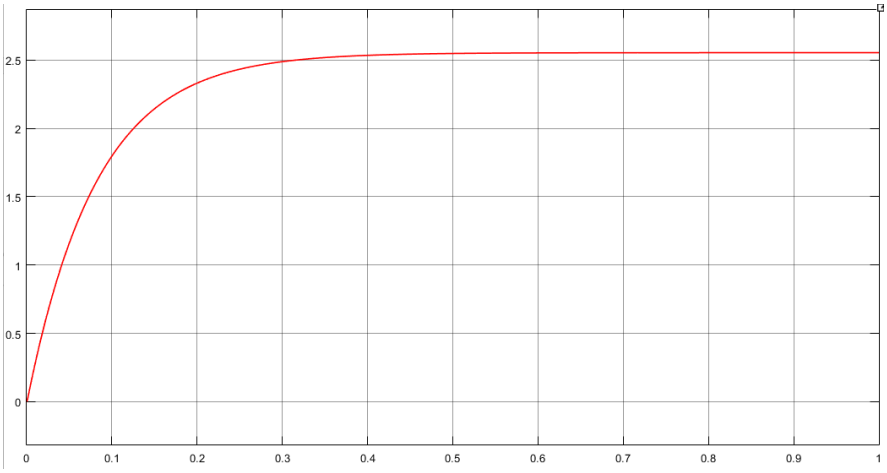

Figura 2-16. Velocidad en rps ante una tensión medida de 10.5 V aplicada al motor

De esta manera, el modelado mediante parámetros experimentales estaría finalizado.

2.2.3. Modelado matemático del motor mediante su respuesta.

Para llevar a cabo este modelado, lo primero que se debe de tener en cuenta es, que, la respuesta de la dinámica que relaciona la velocidad del motor con la tensión de excitación del mismo, tal y como se puede observar en la Figura 2-16, se puede aproximar a un sistema de primer orden [13].

De tal manera, la expresión que define un sistema de primer orden es la Ecuación 46.

$$\frac{Y(s)}{U(s)} = \frac{k}{\tau * s + 1}$$

Ecuación 46

El tipo de entrada que se va a tratar va a ser el de tipo escalón, de esta manera, despejando Y(s) de la Ecuación 46, se obtiene la Ecuación 47.

$$Y(s) = \frac{k}{\tau * s + 1} * \frac{u}{s}$$

Ecuación 47

La Ecuación 47 puede ser transformada, obteniendo así:

$$Y(s) = k * u * \left(\frac{1}{s} - \frac{1}{s + \frac{1}{\tau}}\right)$$

Ecuación 48

Aplicando la transformada inversa de LaPlace a la Ecuación 48.

$$y(t) = k * u * \left(1 - e^{-t/\tau}\right)$$

Ecuación 49

A partir de la Ecuación 49 se pueden extraer los valores necesarios para obtener la dinámica de un sistema de primer orden. Primero se obtendrá el valor de la ganancia de la dinámica, que viene dado por la Ecuación 50.

$$k = \frac{\Delta y}{\Delta u}$$

Ecuación 50

Para obtener la constante de tiempo de una dinámica de primer orden, basta con hacer t=τ en la Ecuación 49.

$$y(\tau) = 0.632 * k * u$$

Ecuación 51

Ahora que se conocen las expresiones necesarias para obtener la dinámica del motor tan sólo es necesario obtener la respuesta en velocidad del mismo, para ello se debe de hacer el mismo ensayo que en la Figura 2-14 pero en este caso se deben omitir el amperímetro y el voltímetro.

De esta manera, excitando el motor con una tensión de 12 V y usando un algoritmo que permite al microcontrolador obtener la velocidad en rps del motor a partir de la salida del encoder, se obtiene la siguiente respuesta.

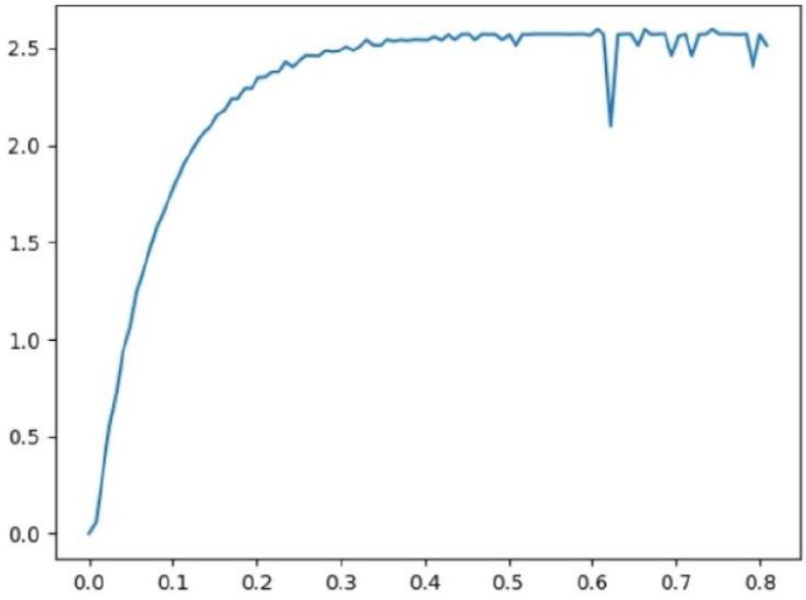

Figura 2-17. Gráfica del ensayo en velocidad del motor en rps en función del tiempo

Sabiendo que la dinámica de la velocidad del motor tiene la siguiente estructura:

$$G_{motor}(s) = \frac{k_m}{\tau_m s + 1}$$

Ecuación 52

A partir de la Figura 2-17, de la Ecuación 50 y la Ecuación 51, se puede obtener la dinámica del motor.

$$G_{motor,vel}(s) = \frac{0.2125}{0.084s + 1}$$

Ecuación 53

A continuación, se realiza la simulación de la Ecuación 53 en Simulink.

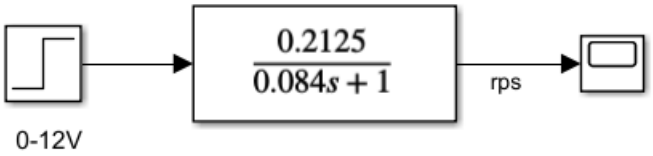

Figura 2-18. Simulación de la dinámica del motor obtenida mediante su respuesta

Obteniendo así la siguiente respuesta:

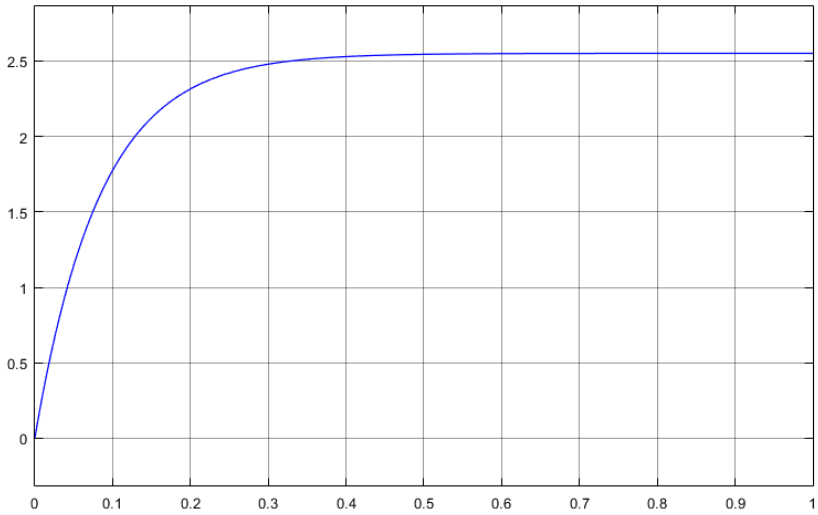

Figura 2-19. Respuesta de la simulación de la dinámica del motor obtenida mediante su respuesta

Esta respuesta ya se encuentra en rps, por lo que conociendo la Ecuación 29, la dinámica del motor que relaciona la posición de este en revoluciones con la tensión de entrada sería:

$$G_{motor}(s) = \frac{0.2125}{s * (0.084s + 1)}$$

Ecuación 54

2.2.4. Comparación entre ambos métodos de modelado.

Una vez obtenidos los resultados de ambos modelados, se deben de comparar para comprobar la validez de ambos métodos.

Para ello se lleva a cabo la simulación de la Figura 2-20.

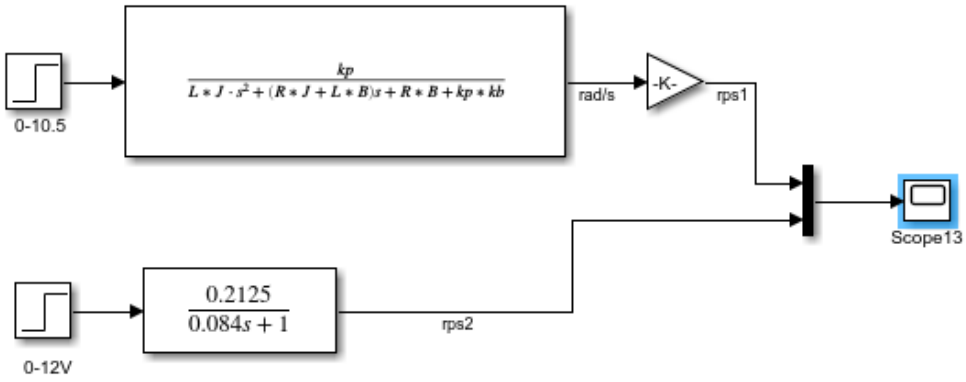

Figura 2-20. Simulación para comparar ambos modelados

Tal y como se observa en la Figura 2-21, las respuestas de ambos modelados son prácticamente idénticas, sin embargo, para hacer una comparación más profunda se va a desarrollar una simplificación de la Ecuación 37.

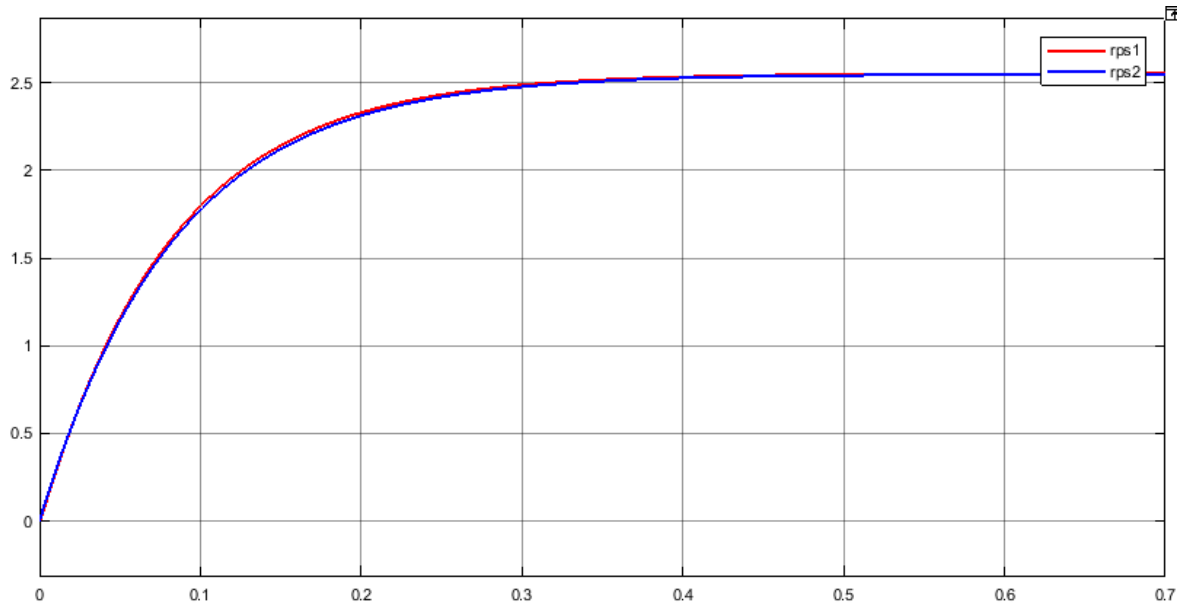

Figura 2-21. Resultados de la simulación para comparar ambos modelados

Para realizar la simplificación mencionada anteriormente, lo primero que se debe de hacer es sustituir los valores de la Tabla 2 en la Ecuación 37, obteniendo así la siguiente expresión:

$$\frac{\Omega(s)}{E(s)} = \frac{0.6157}{1.73 * 10^{-5} * s^2 + 0.0332 * s + 0.4031}$$

Ecuación 55

Los polos de la dinámica de la Ecuación 55 son los siguientes:

$$p_1 = -1906.86;$$

$$p_2 = -12.22;$$

Como se cumple la Ecuación 56, se puede aplicar reducción de polo dominante.

$$p_1 \leq 5 * p_2$$

Ecuación 56

Eliminando el polo más alejado del origen y respetando la ganancia estática de la dinámica de la Ecuación 55, se obtiene la siguiente expresión:

$$\frac{\Omega(s)}{E(s)} = \frac{1.5274}{0.0818 * s + 1}\left(\frac{rad}{s}\right)$$

Ecuación 57

Para poder comparar la expresión obtenida con la dinámica que se consiguió a través de la respuesta del motor, tan sólo se debe de aplicar a la Ecuación 57 la relación lineal que aparece en la Figura 2-21, obteniendo así la velocidad en rps. De esta manera, la expresión resultante es la siguiente:

$$\frac{\Omega(s)}{E(s)} = \frac{0.2431}{0.0818 * s + 1}(rps)$$

Ecuación 58

Comparando la Ecuación 58 con la Ecuación 53, se puede afirmar que ambas dinámicas son casi idénticas tanto gráfica como matemáticamente. De aquí en adelante, debido a la simplicidad de su obtención se usará la dinámica de la Ecuación 53.

2.3. Relación entre el ángulo de la barra y el ángulo del motor.

Para finalizar el modelado de dinámicas, se desea obtener la relación que existe entre la posición del motor en revoluciones y el ángulo de giro de la barra en radianes, para ello se hará uso de relaciones lineales, tal y como se puede observar en la Figura 2-1.

La primera que se debe de obtener es G_1, esta relación lineal permite obtener $\theta_2(t)$, que es la posición del motor en radianes, a partir de $\theta_1(t)$, que es la posición del motor en revoluciones. Obteniendo así la siguiente expresión:

$$G_1 = \frac{\theta_2(s)}{\theta_1(s)} = 2 * \pi$$

Ecuación 59

La relación lineal G_2, permite obtener $\alpha(t)$, que es el grado de inclinación de la barra, a partir de $\theta_2(t)$.

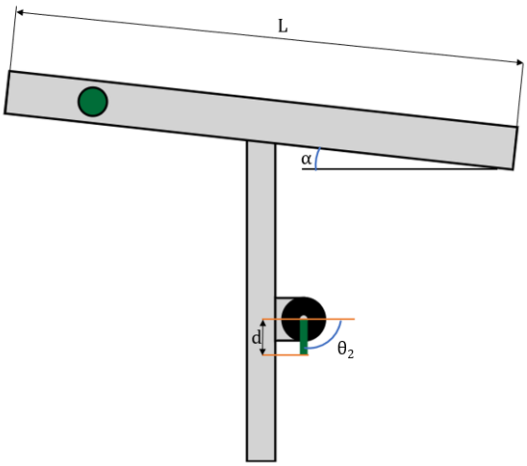

Figura 2-22. Esquema de la barra y del motor con variables necesarias para obtener G_2

Observando la Figura 2-22, se obtiene la siguiente expresión:

$$\alpha = \frac{d}{L} * \theta_2$$

Ecuación 60

Siendo:

- d, offset del brazo del motor.
- L, longitud de la barra.

De esta manera, se obtiene $G_2(s)$:

$$G_2(s) = \frac{\alpha(s)}{\theta_2(s)} = \frac{d}{L}$$

Ecuación 61

Sabiendo que la barra tiene una longitud de 50 cm y que el brazo del motor tiene una longitud de 3.5 cm, el valor de $G_2(s)$ será:

$$G_2(s) = 0.07$$

Ecuación 62

3. Maqueta.

En este capítulo, tras haber tratado el modelado de sistemas, se tratan los distintos dispositivos que conforman la planta real que posteriormente se usará para llevar a cabo el laboratorio remoto, al igual que el conexionado entre estos.

3.1. Elementos hardware.

En este apartado se desarrollan los distintos elementos hardware que constituyen la maqueta, además de su respectivo funcionamiento, estos elementos hardware son un motor de corriente continua, una Raspberry Pi 4 Model B, una cámara Raspberry v2.1, un panel led de iluminación, un puente H, un módulo de relés y por último un adaptador AC.

3.1.1. Motor CC.

Se trata de un motor de corriente continua [14], con escobillas de 12 V y con una caja de engranajes con una relación 70:1, tal y como ya se ha mencionado en el capítulo del Modelado del motor.

Figura 3-1. Caja de engranajes del motor

Como se puede observar en la Figura 3-1, la caja de engranajes está formada en su mayoría por engranajes rectos, aunque también se pueden observar helicoidales, cuya función será reducir el ruido producido por el giro del motor y mejorar su eficiencia.

El motor además tiene incorporado un encoder de efecto Hall y dos canales cuya finalidad es percibir la rotación del motor. Además, tiene una resolución de 64 CPR (counts per revolution) lo que sumado al dato de que su caja de engranajes tiene una relación de 70:1 permite conocer que una revolución equivaldrá a 4480 pulsos.

El encoder tiene 6 cables, cuyas funciones ya han sido detalladas en la Tabla 1, pero es importante centrarse en las salidas de dicho encoder, que se corresponde con los cables amarillo y blanco, cuya salida puede ser representada a través de un osciloscopio.

Figura 3-2. Salidas del encoder vistas en un osciloscopio

En la Figura 3-2, se puede visualizar las salidas del encoder, que son señales cuadradas las cuales están desfasadas aproximadamente 90º. La frecuencia de las transiciones indicará la velocidad del motor, mientras que el orden de dichas señales indicará la dirección de giro del mismo.

Por último, es importante dejar claro que, para esta maqueta, el motor tiene la función de actuador, ya que será el encargado de manipular el ángulo de inclinación de la barra.

3.1.2. Raspberry Pi.

Es considerado un ordenador monoplaca destinado a usos docentes, el cual fue desarrollado para promover y enseñar electrónica. A pesar de esto, es una excelente opción para este tipo de proyectos debido a sus altas prestaciones [15].

Para esta maqueta, se usa el modelo de la Figura 3-3. Una de las principales características de este modelo son sus 40 pines que conforman el encabezado GPIO, de los cuales, la mayoría pueden ser configurados para cumplir función de entrada o salida.

Figura 3-3. Raspberry Pi 4 Model B

En la Figura 3-4 se pueden observar los 40 pines de los que consta la Raspberry además de su respectiva numeración.

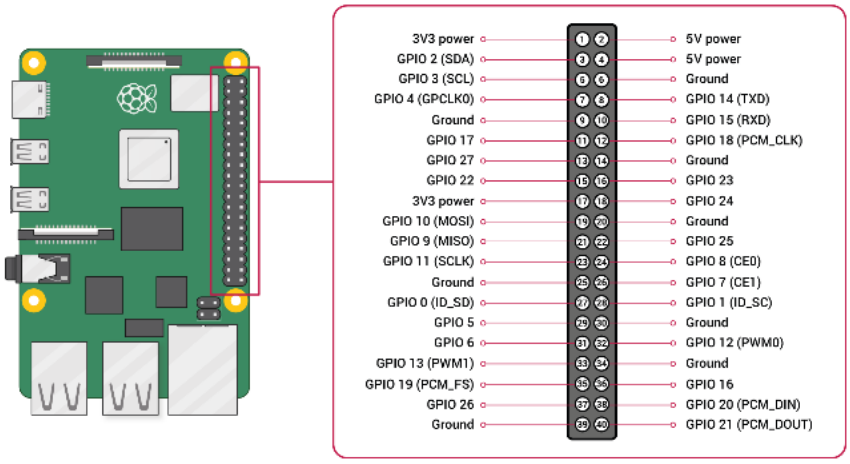

Figura 3-4. GPIO de la Raspberry

En este caso, este dispositivo será usado como microcontrolador, permitiendo así realizar el control de la planta además de poder recibir señales como las salidas del encoder y de poder actuar sobre otros dispositivos de manera indirecta, como puede ser el Panel led de iluminación.

3.1.3. Cámara Raspberry v2.1

Esta cámara, que tiene una resolución de 8 MP, es de los mismos desarrolladores de la Raspberry Pi mencionada en el punto anterior. Por lo que esta puede ser conectada directamente al ordenador monoplaca, esto se puede observar en la Figura 3-5.

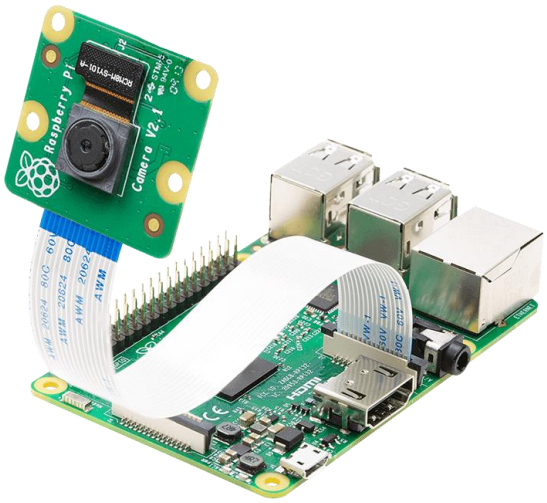

Figura 3-5. Cámara Raspberry v2.1 conectado a Raspberry Pi 4 Model B

La función que tiene este dispositivo es la de medir la posición de la bola mediante un algoritmo que llevará a cabo el microcontrolador.

Un factor muy importante a tener en cuenta es que la distancia focal de esta cámara es de 3.04 mm, por lo que se puede asegurar que la imagen no tendrá efecto ojo de pez, lo cual produciría complicaciones para medir la posición de la bola.

3.1.4. Panel led de iluminación.

Se trata de un panel led con unas medidas de 30x60 cm y una potencia de 24 W. En la Figura 3-6 se puede observar el panel usado para la maqueta.

Figura 3-6. Panel led de iluminación

Su función es la de facilitar la detección de la posición de la bola por parte de la cámara, ya que, si no se usa este panel de iluminación, el procesado de la imagen se complica en gran medida, tal y como se puede ver en la Figura 3-7.

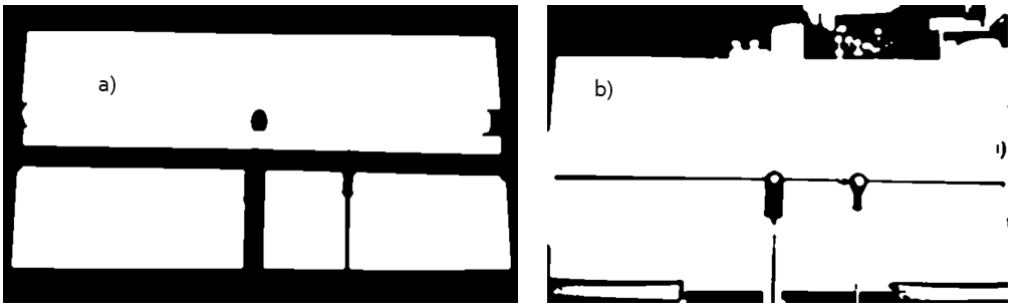

Figura 3-7. Procesamiento de la imagen. a) Panel de iluminación encendido. b) Panel de iluminación apagado

3.1.5. Puente H.

Se trata de un dispositivo electrónico cuya finalidad será la de excitar al motor con la tensión que este necesita, para así conseguir manipular variables como son la velocidad y sentido de giro del motor.

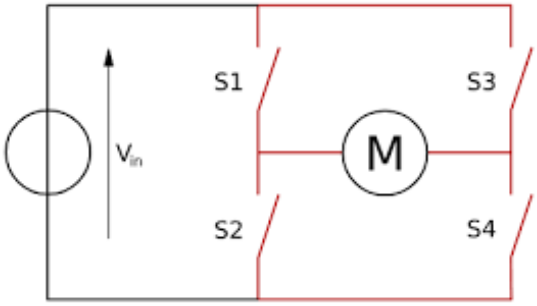

Figura 3-8. Circuito simplificado del funcionamiento del Puente H

En la Figura 3-8, se puede observar el funcionamiento simplificado de un puente H, de esta manera se llega a la conclusión que si los interruptores S_1 y S_4 se cierran el motor girará en un sentido, mientras que si son los interruptores S_2 y S_3 los que se cierran el motor girará en el otro sentido.

El puente H seleccionado se trata de un modelo Dual H-Bridge V1.3 [16], este se puede visualizar en la Figura 3-9. Se trata de una versión mejorada del modelo DF-MDV1.0, ya que puede funcionar durante más tiempo debido a un mayor disipador térmico.

Figura 3-9. Dual H-Bridge V1.3

Se adjunta un esquema donde se pueden visualizar todas conexiones que tiene este dispositivo en la Figura 3-10 además de la Tabla 3 con las diferentes posibilidades para excitar el motor.

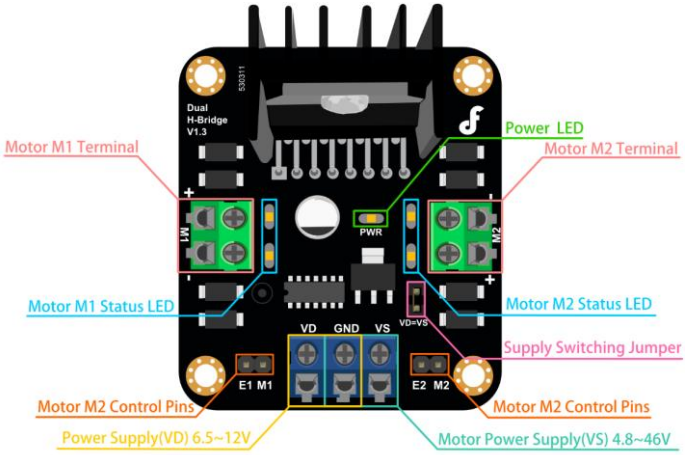

Figura 3-10. Conexiones presentes en el Puente H

E	M	RUN
LOW	LOW/HIGH	STOP
HIGH	HIGH	Back Direction
HIGH	LOW	Forward direction
PWM	LOW/HIGH	Speed

Tabla 3. Combinaciones para excitar el motor

Es muy importante tener en cuenta que, para esta maqueta, el puente de conmutación de suministro (Supply Switching Jumper), estará cortocircuitado, de esta manera se consigue que V_D=V_S.

3.1.6. Relé.

Para esta maqueta, se utiliza un módulo de 4 relés de 5 V de la compañía Songle. Este módulo se puede controlar con un microcontrolador como es el caso de la Raspberry Pi.

Este módulo posee 4 entradas para cada uno de los relés y dos pines, uno de alimentación V_{CC} y otro de tierra GND. Además, es importante tener en cuenta que según el tipo de conexión que se realice con el cableado del dispositivo que se vaya a excitar, este puede encontrarse normalmente abierto o normalmente cerrado.

Figura 3-11. Módulo de 4 Relés

3.1.7. Adaptador AC.

Será el dispositivo encargado de convertir la tensión alterna en tensión continua para así poder alimentar al puente H que será el encargado de excitar el motor.

Para la maqueta, se ha utilizado un adaptador de alterna el cual convierte una tensión de entrada de entre 100 y 240 V y una frecuencia de 50-60 Hz a una tensión continua de 12 V.

Figura 3-12. Adaptador AC

3.2. Conexionado eléctrico de elementos.

En la Figura 3-13 se puede observar el conexionado de todos los elementos hardware usados en la maqueta.

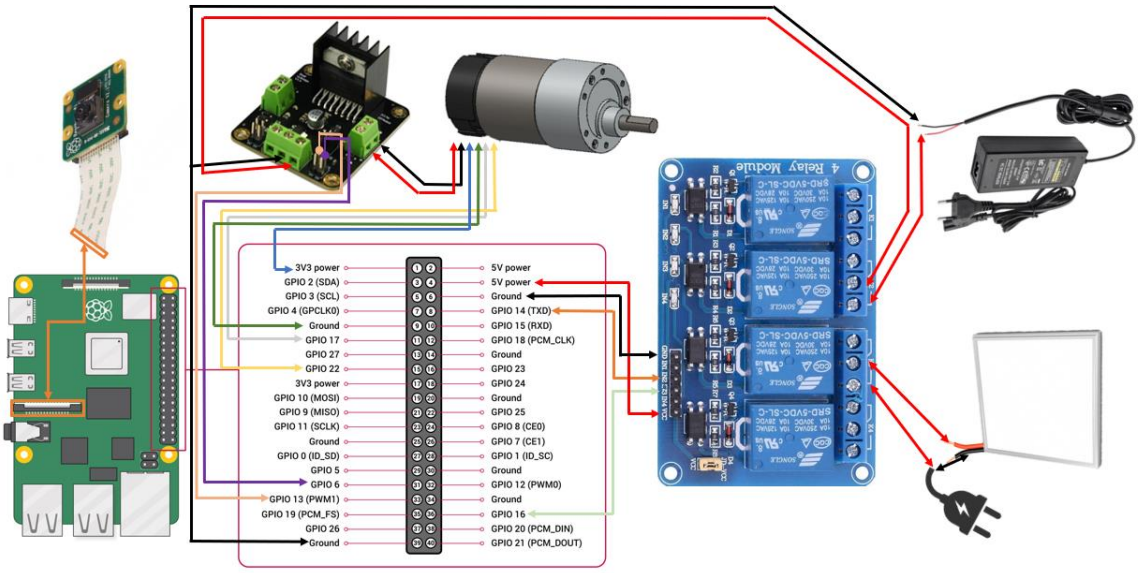

Figura 3-13. Conexionado de los dispositivos electrónicos que forman la maqueta

Tal y como se puede observar en el esquema de conexiones:

- La alimentación del puente H y del panel de iluminación son controlados por el módulo de relés, siendo este módulo a su vez manipulado por la Raspberry Pi mediante los pines 8 y 16.
- El puente H es manipulado a través de la Raspberry Pi mediante los pines 31 y 33, siendo el pin 33 una señal PWM.
- Las salidas del encoder serán recibidas en la Raspberry Pi mediante los pines 11 y 15 para posteriormente ser procesadas.
- La cámara va conectada directamente a la Raspberry Pi y fijada a un perfil de aluminio para estar frente al sistema barra-bola y así poder cumplir con su cometido, tal como se puede observar en la Figura 3-14.

Figura 3-14. Colocación de la cámara Raspberry Pi

3.3. Conexión mecánica entre motor y barra.

A continuación, se debe de abordar cómo se lleva a cabo la transmisión del movimiento entre motor y barra.

Para ello la barra, ha sido situada sobre un perfil que a su vez se encuentra sobre un eje de giro. Al eje de giro del motor, se le ha añadido un brazo como el mostrado en la Figura 3-15.

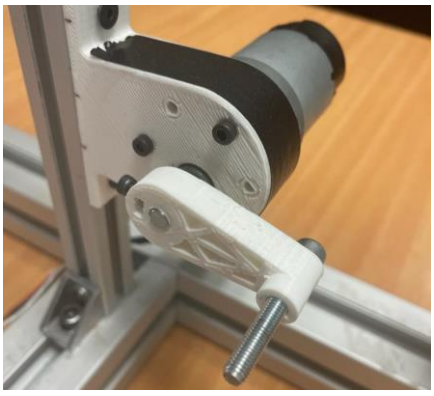

Figura 3-15. Brazo del motor

Finalmente, para unir el perfil que sostiene la barra y el brazo del motor, se ha usado un eje con rótulas de baja fricción para intentar disminuir las fuerzas de rozamiento.

Figura 3-16. Rótula de baja fricción

El resultado final de esta parte de la maqueta se puede visualizar en la Figura 3-17.

Figura 3-17. Unión del motor con la barra

4. Lazo de control.

En este capítulo se aborda qué lazo de control se emplea además de los distintos parámetros que requerirá el controlador para poder llevar a cabo el control de la planta, todo basándose en las dinámicas previamente obtenidas y en herramientas como el lugar de las raíces y simulaciones en Simulink.

4.1. Controlador PID.

Lo primero que se debe de tener claro es que la dinámica de control, es aquella que tiene como entrada un error, e(t) y como salida una variable controlada, m(t).

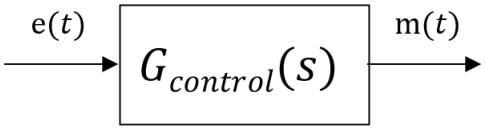

Figura 4-1. Dinámica del control genérica

A continuación, se debe de diferenciar qué tipos de acciones puede tener un controlador [17], de esta manera se puede diferenciar entre:

- Acción proporcional: viene dada por la Ecuación 63.

$$m(t) = k_c * e(t)$$

Ecuación 63

Siendo k_c la ganancia proporcional, ya que esta provocará una acción de control sobre el actuador en función del error recibido.

La principal función que tiene esta función es la de aumentar la velocidad de respuesta del sistema.

Sin embargo, esta ganancia proporcional puede ser acotada por dos factores, el primero son los límites físicos del actuador y el segundo son el lugar de las raíces, ya que puede existir una $k_{c,última}$, la cual si es sobrepasada haga que el sistema se vuelva inestable.

- Acción integral: viene dada por la Ecuación 64.

$$m(t) = \frac{k_c}{T_i} * \int e(t)\, dt$$

Ecuación 64

La integral del error es la magnitud que continúa creciendo aun manteniéndose constante el error, por lo que mientras exista error, la acción integral evolucionará.

Se usa junto a la acción proporcional, ya que por sí sola, la respuesta del sistema sería muy lenta.

La finalidad de esta acción es aumentar el tipo del sistema realimentado, asegurando así que el error estacionario sea nulo frente a una entrada escalón.

T_i se define como el tiempo en que la acción integral iguala a la acción proporcional a error constante, tal y como se puede observar en la Figura 4-2.

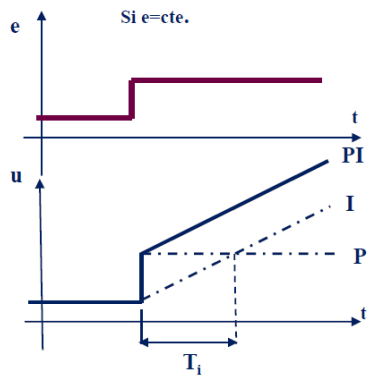

Figura 4-2. Evolución de variable controlada ante error constante con un PI

Por último, a mayor acción integral, menor será T_i y mayor será la velocidad de la eliminación del error.

- Acción derivativa: viene dada por la Ecuación 65.

$$m(t) = k_c * T_d * \frac{de(t)}{dt}$$

Ecuación 65

El valor de la derivada de una función indica cómo cambia esa función a lo largo del tiempo. Por lo que esta acción se puede definir como una acción anticipativa.

Su finalidad es que la respuesta sobreoscile menos, aumentando así la velocidad de respuesta del sistema, pero no influye sobre el régimen estacionario.

T_d se define como el tiempo en que la acción proporcional iguala a la acción derivativa con el error variando linealmente, tal y como se puede observar en la Figura 4-3.

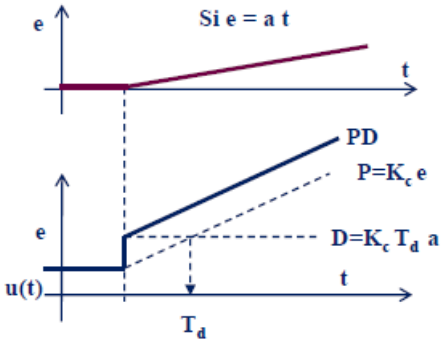

Figura 4-3. Evolución de variable controlada ante error lineal con un PD

Ahora que se conocen las distintas acciones que se pueden encontrar en un controlador PI, se pueden definir los distintos tipos de controlador que se pueden encontrar.

- Controlador Proporcional Derivativo (PD).

$$G_{control}(s) = k_c * (1 + T_d * s)$$

Ecuación 66

- Controlador Proporcional Integral (PI).

$$G_{control}(s) = k_c * \left(1 + \frac{1}{T_i * s}\right)$$

Ecuación 67

- Controlador Proporcional Integral Derivativo (PID).

$$G_{control}(s) = k_c * \left(1 + T_d * s + \frac{1}{T_i * s}\right)$$

Ecuación 68

4.2. Lazo de control elegido.

Una vez conocidos los distintos tipos de controlador que se pueden usar, se debe de llevar a cabo el lazo de control. Para ello, se puede tomar de partida el bucle abierto de la Figura 2-1. Sin embargo, al bucle abierto de dicha figura, se le debe de aplicar una realimentación y dos bloques que no se contemplaron en el modelado del motor, siendo estos el bloque PWM y el del DRIVER, obteniendo así el lazo de control de la Figura 4-4.

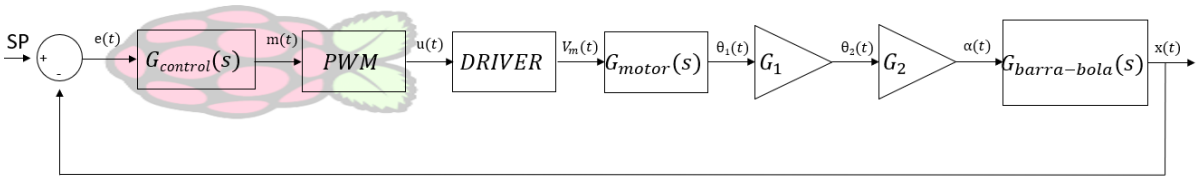

Figura 4-4. Primera opción de lazo de control

Antes de continuar con el estudio de este lazo de control, se deben de obtener las dinámicas PWM y DRIVER.

Tal y como se observa en la anterior figura, tanto el control como la generación de la señal PWM son tareas que lleva a cabo la Raspberry Pi.

De esta manera y siguiendo el lazo, el error, e(t), es la entrada de la dinámica del control, obteniendo como salida la señal m(t). Es importante tener claro que m(t) no es la variable manipulada, será gracias al bloque PWM y DRIVER, que la señal m(t) se adaptará y transmitirá hasta obtener $V_m(t)$ que sí es la variable manipulada.

El valor de m(t) indica el % de tiempo que una señal periódica se encuentra en su valor máximo (Duty Cycle), por lo que dicho valor podrá oscilar entre 0 y 100%. La

Figura 4-5 muestra varios ejemplos. Véase que, si se tiene un Duty Cycle de 50%, la señal se encontrará la mitad del periodo a nivel alto y la otra mitad a nivel bajo.

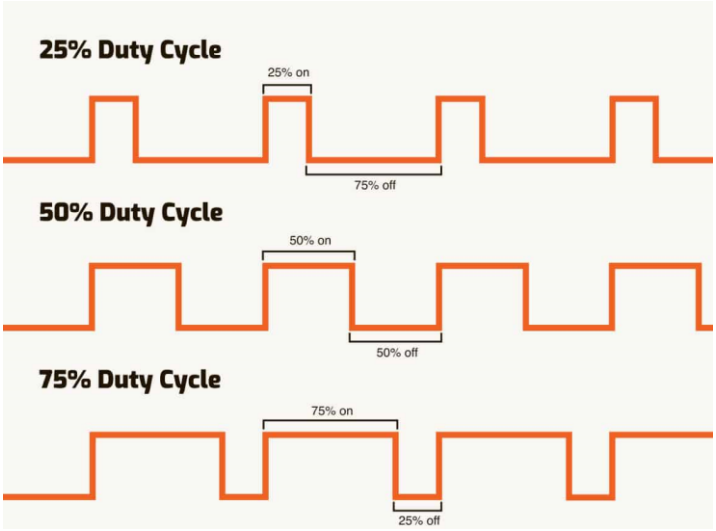

Figura 4-5. Ejemplo de valores de la variable Duty Cycle

Como ya se sabe, el motor debe de ser excitado con tensión (0-12V), por lo que la señal m(t) debe de ser transmitida y transformada.

Para que una señal pueda ser transmitida tiene que ser modulada y esto es precisamente lo que realiza el bloque PWM. Dicho bloque recibe el Ducy Cycle y obtiene el promedio de la tensión de salida a lo largo del tiempo, siendo este la señal u(t) de la Figura 4-4. El valor de dicha tensión puede ser como máximo de 3.3 V ya que es lo que permite generar la fuente de tensión de la Raspberry.

Dicha tensión no es compatible con la tensión necesaria para excitar el motor DC, de ahí la necesidad de incorporar un Driver. Dicho driver escala la tensión que recibe a la tensión necesaria para excitar el motor, obteniendo así la señal $V_m(t)$. La Figura 4-6 muestra un ejemplo.

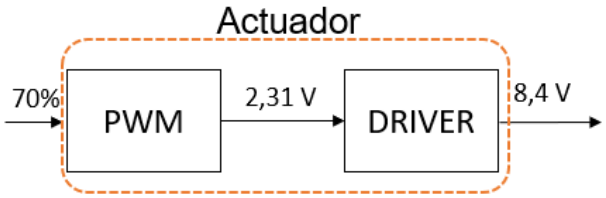

Figura 4-6. Ejemplo transmisión y escala de la tensión al motor DC

De esta manera, se obtienen las dinámicas PWM y DRIVER, que servirán tanto para la primera opción planteada en la Figura 4-4, como para cualquier otro posible lazo de control.

$$G_{PWM}(s) = \frac{3.3}{100}\left(\frac{v}{\%}\right)$$

Ecuación 69

$$G_{Driver}(s) = \frac{12}{3.3}\left(\frac{v}{v}\right)$$

Ecuación 70

De esta manera, se puede continuar con el estudio del lazo de control de la Figura 4-4. Para ello, es necesario recordar la Ecuación 53, la Ecuación 21, la Ecuación 59 y la Ecuación 60. De esta manera, y aplicando álgebra de bloques, se obtendrá una dinámica la cual tiene un polo doble en el origen y dos polos en el semiplano izquierdo.

El hecho de tener un polo doble en el origen, hace que usar un controlador PID, no sea una opción muy acertada, debido a que producirá que el sistema se vuelva inestable, además que tendría una complejidad elevada para poder ser implementado en código.

Todos estos motivos, llevan a la conclusión de que se debe de intentar usar otro tipo de lazo de control. Siendo en este caso, un control en cascada [18]. Las condiciones que indican que este tipo de control es buena opción son las siguientes:

- El proceso que se desea controlar está formado por diferentes etapas.
- Se tiene una variable manipulable y más de una variable medida.
- Las perturbaciones afectan directamente a la variable de proceso manipulada.
- La variable controlada es mucho más lenta que las intermediarias.

De esta manera, el nuevo lazo de control queda reflejado en la Figura 4-7. Pudiendo ver así que ahora existen dos controladores:

- $G_{c1}(s)$: control esclavo, encargado de realizar el control en posición del motor.
- $G_{c2}(s)$: control maestro, encargado de realizar el control del sistema barra-bola.

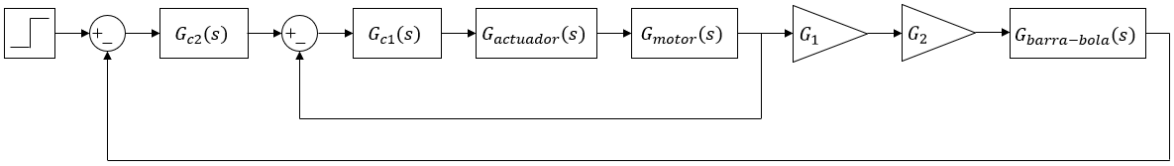

Figura 4-7. Lazo de control en cascada

Se debe además de tener cuenta que la dinámica denominada $G_{actuador}(s)$ es la unión de la dinámica PWM y del DRIVER, por lo que recurriendo a la Ecuación 69 y a la Ecuación 70, se puede obtener la siguiente expresión:

$$G_{actuador}(s) = \frac{3.3}{100} * \frac{12}{3.3} = 0.12$$

Ecuación 71

4.3. Cálculos para obtener respuesta críticamente amortiguada.

En este apartado se llevarán a cabo los cálculos necesarios para obtener una respuesta críticamente amortiguada del sistema. Para ello, el primer control que se debe de calcular es el esclavo.

4.3.1. Control esclavo.

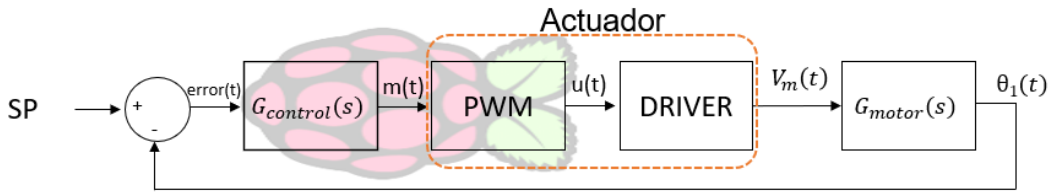

Figura 4-8. Lazo de control esclavo

Lo primero que se debe de obtener es la dinámica de bucle abierto.

$$G_{ba}(s) = G_{control}(s) * G_{actuador}(s) * G_{motor}(s)$$

Ecuación 72

Sustituyendo la Ecuación 71 y la Ecuación 53 en la Ecuación 72.

$$G_{ba}(s) = G_{control}(s) * \frac{0.0255}{s * (0.084 * s + 1)}$$

Ecuación 73

El siguiente paso, es elegir qué tipo de controlador se va a llevar a cabo, y a pesar de que con un controlador proporcional se puede obtener una respuesta críticamente amortiguada, la mejor opción va a ser usar un controlador proporcional derivativo, ya que, de esta forma, se conseguirá reducir las posibles oscilaciones del motor. El uso de una acción integral está descartado, debido a que se trata de un sistema con un polo en el origen, por lo que se puede garantizar que el error ya va a ser nulo. Por lo que la Ecuación 73 puede transformarse en la siguiente expresión.

$$G_{ba}(s) = k_c * (1 + T_d * s) * \frac{0.0255}{s * (0.084 * s + 1)}$$

Ecuación 74

Para obtener los parámetros necesarios de este controlador, se usará el método de cancelación cero-polo.

$$G_{ba}(s) = k_{c*} * T_d * \left(s + \frac{1}{T_d}\right) * \frac{0.303}{s * (s + 11.9)}$$

Ecuación 75

De donde podemos obtener el valor de T_d.

$$T_d = 0.084$$

Ecuación 76

Sustituyendo el valor de T_d en la Ecuación 75, se obtendrá:

$$G_{ba}(s) = k_{c*} * \frac{0.0255}{s}$$

Ecuación 77

Para obtener el valor de la acción proporcional, primero hay que obtener la dinámica de bucle cerrado en la Ecuación 77, obteniendo así la Ecuación 78.

$$G_{bc}(s) = \frac{0.0255 * k_c}{s + 0.0255 * k_c}$$

Ecuación 78

Tras una serie de pruebas en el laboratorio, se determinó que la respuesta óptima del motor para después poder ser usado en el sistema barra-bola, debe de tener un tiempo de establecimiento de 3 segundos. Para ello, primero se transforma la Ecuación 78.

$$G_{bc}(s) = \frac{1}{\frac{s}{0.0255 * k_c} + 1}$$

Ecuación 79

El tiempo de establecimiento viene dado por la Ecuación 80.

$$t_s = 4 * \tau$$

Ecuación 80

Por lo que con la Ecuación 79 y la Ecuación 80, se obtiene:

$$4 * \frac{1}{0.0255 * k_c} = 3$$

Ecuación 81

De donde se puede obtener el valor de la acción proporcional, siendo este:

$$k_c = 52.28$$

Ecuación 82

Obteniendo así la dinámica del control esclavo, siendo:

$$G_{c1}(s) = 52.28 * (1 + 0.084 * s)$$

Ecuación 83

4.3.2. Simulación del lazo de control esclavo.

Para comprobar que los cálculos se han realizado correctamente se realizará la siguiente simulación en Simulink, donde la referencia será de una revolución y el bloque del controlador PID tendrá la misma expresión que la Ecuación 83.

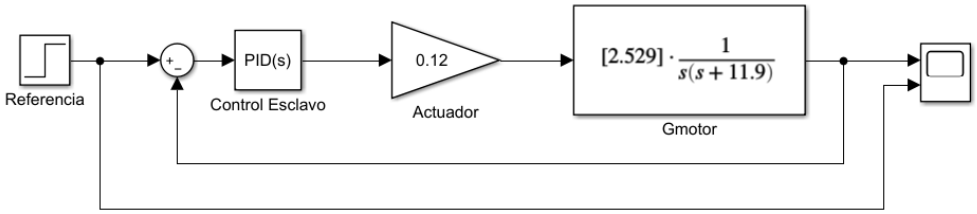

Figura 4-9. Simulación del lazo del control esclavo

Obteniendo a partir de dicha simulación los siguientes resultados.

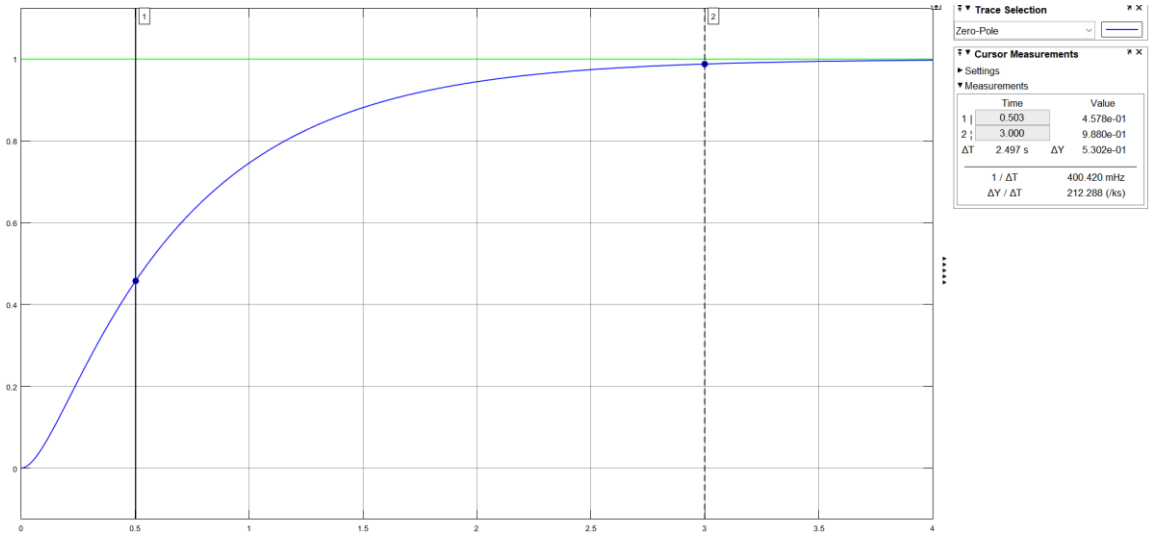

Figura 4-10. Resultados de la simulación del lazo del control esclavo

Tal y como se puede observar en la Figura 4-10, el error en régimen estacionario es nulo, el sistema es críticamente amortiguado y, además, se ha conseguido el tiempo de establecimiento deseado.

4.3.3. Control maestro.

El primer paso que se debe de llevar a cabo es sustituir el lazo del control esclavo en el lazo de control maestro, para ello se usará la Ecuación 78 y la Ecuación 82, obteniéndose así la siguiente expresión.

$$G_{LazoInterno}(s) = \frac{1.33}{s + 1.33}$$

Ecuación 84

Quedando el lazo de control maestro de la siguiente manera:

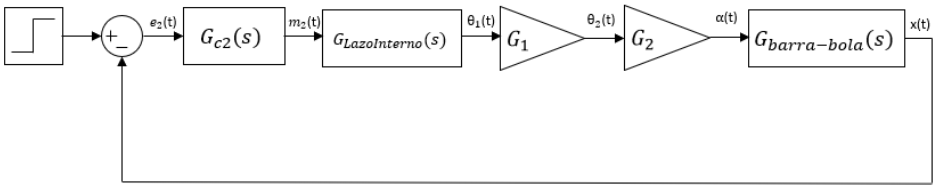

Figura 4-11. Lazo de control maestro con la simplificación del lazo interno

A continuación, se calcula la dinámica de bucle abierto, obteniéndose así la siguiente expresión:

$$G_{ba}(s) = G_{c2}(s) * G_{LazoInterno}(s) * G_1 * G_2 * G_{barra-bola}(s)$$

Ecuación 85

Sustituyendo las distintas expresiones necesarias en la Ecuación 85, se obtiene:

$$G_{ba}(s) = G_{c2}(s) * \frac{4.094}{s * (s + 1.33) * (s + 2.814)}$$

Ecuación 86

El siguiente paso es seleccionar qué tipo de controlador se va a usar, al igual que en el control esclavo, se va a usar un PD, debido a que es la opción que ofrece una respuesta más rápida y que además permite obtener un sistema críticamente amortiguado. La acción integral, al igual que en el control esclavo está descartada, ya que ya se tiene un polo en el origen el cual garantiza un error nulo. De esta manera se obtiene la siguiente expresión.

$$G_{ba}(s) = k_{c*} * T_d * \left(s + \frac{1}{T_d}\right) * \frac{4.094}{s * (s + 1.33) * (s + 2.814)}$$

Ecuación 87

Mediante el método de cancelación cero-polo, se obtiene el valor de T_d, quedando además la Ecuación 87 simplificada.

$$T_d = 0.75$$

Ecuación 88

$$G_{ba}(s) = k_c * \frac{3.07}{s * (s + 2.814)}$$

Ecuación 89

El último parámetro que queda por obtener es el de la acción proporcional, para ello, primero se debe de obtener la dinámica del bucle cerrado.

$$G_{bc}(s) = \frac{3.07 * k_c}{s * (s + 2.814) + 3.07 * k_c}$$

Ecuación 90

Observando el lugar de las raíces de la dinámica de la Ecuación 90, se puede observar que para que el sistema sea críticamente amortiguado, ambos polos deben de encontrarse en s=-1.407.

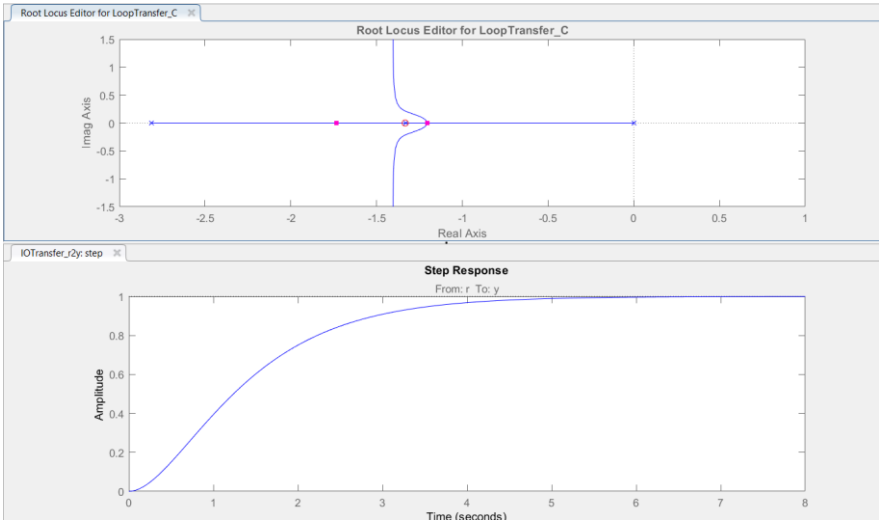

Figura 4-12. Lugar de las raíces

A partir de estos datos, se puede obtener la siguiente ecuación:

$$s * (s + 2.814) + 3.07 * k_c = (s + 1.407)^2$$

Ecuación 91

De donde se puede despejar el valor de la acción proporcional, siendo este de:

$$k_c = 0.645$$

Ecuación 92

Obteniendo así la dinámica del control maestro, siendo:

$$G_{c2}(s) = 0.645 * (1 + 0.75 * s)$$

Ecuación 93

4.3.4. Simulación del lazo de control maestro.

Para comprobar que los cálculos se han realizado correctamente se realizará la siguiente simulación en Simulink, donde la referencia será una entrada escalón unitario y el bloque del controlador PID tendrá la misma expresión que la Ecuación 93.

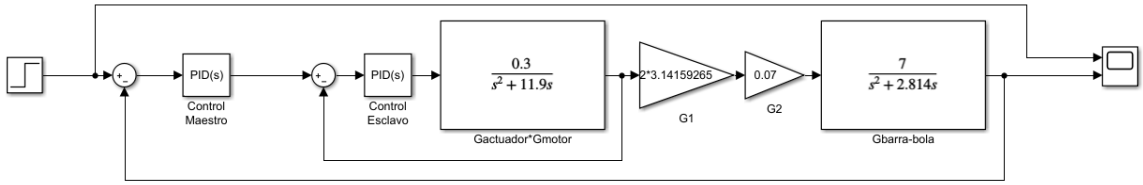

Figura 4-13. Simulación del lazo maestro

Obteniendo así la siguiente respuesta:

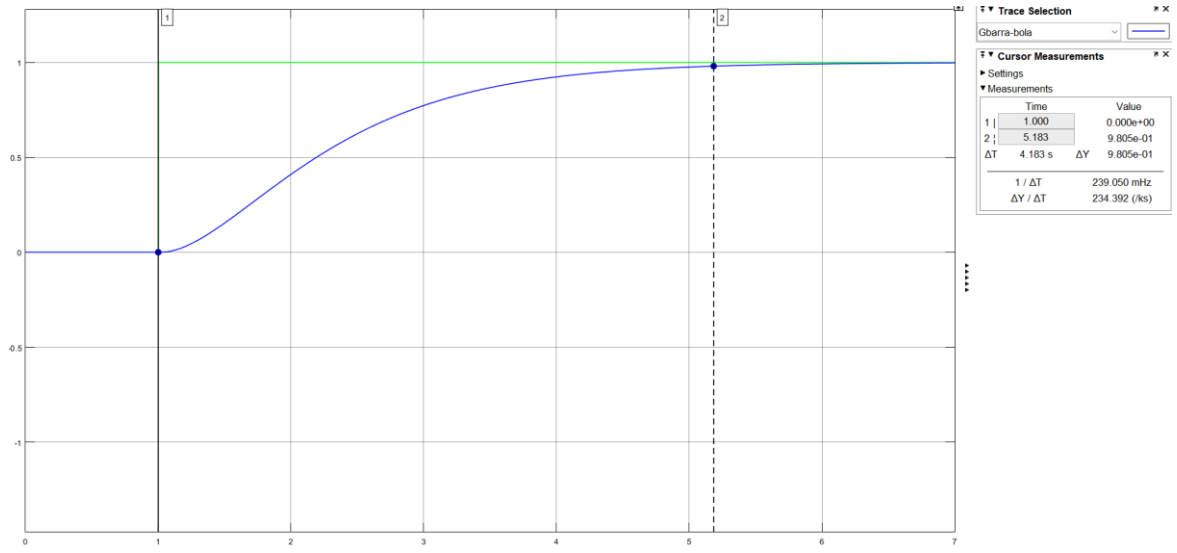

Figura 4-14. Resultado de la simulación del lazo maestro

Tal y como se puede observar en la Figura 4-14, la respuesta es críticamente amortiguada y el tiempo de establecimiento se encuentra en torno a 4.183 s.

4.4. Límites físicos existentes en la planta real.

Para finalizar este capítulo, se desea poner de manifiesto los límites físicos que existen en la planta real y que no han sido reflejados en la simulación de la Figura 4-14. El objetivo es que los resultados obtenidos en simulación, sean lo más cercanos posibles a los que se obtendrán en la realidad. Para ello, se hará uso de un elemento disponible en Simulink denominado saturador.

Antes de llevar a cabo la simulación, se debe de tener en cuenta cuáles son los límites físicos existentes:

- Duty Cycle: como se ha comentado anteriormente, este puede tener un valor máximo del 100%.
- Grado de giro del motor: si se considera que el motor se encuentra a 0 grados cuando la barra se encuentra totalmente horizontal, el motor no puede girar más de 90 grados en ambas direcciones.

Por lo que aplicando dichos saturadores a la simulación de la Figura 4-13, se puede llevar a cabo una comparación entre los resultados con límites físicos y sin límites físicos.

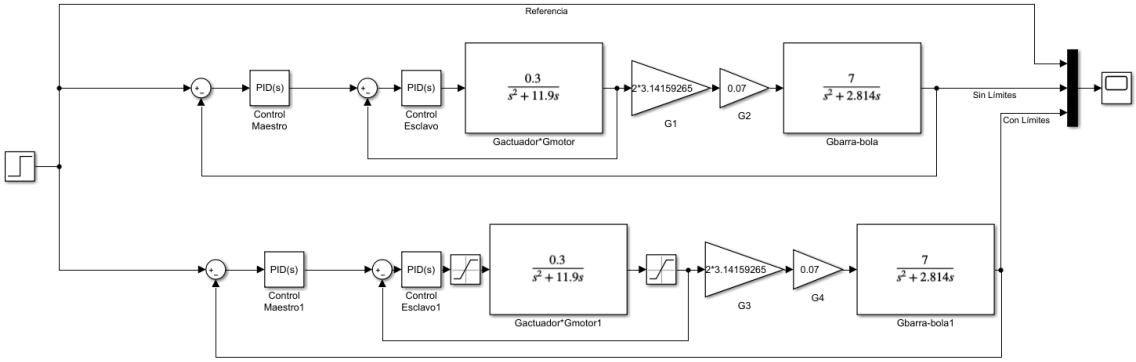

Figura 4-15. Simulación para comparar efectos de los límites físicos

Obteniendo así los siguientes resultados:

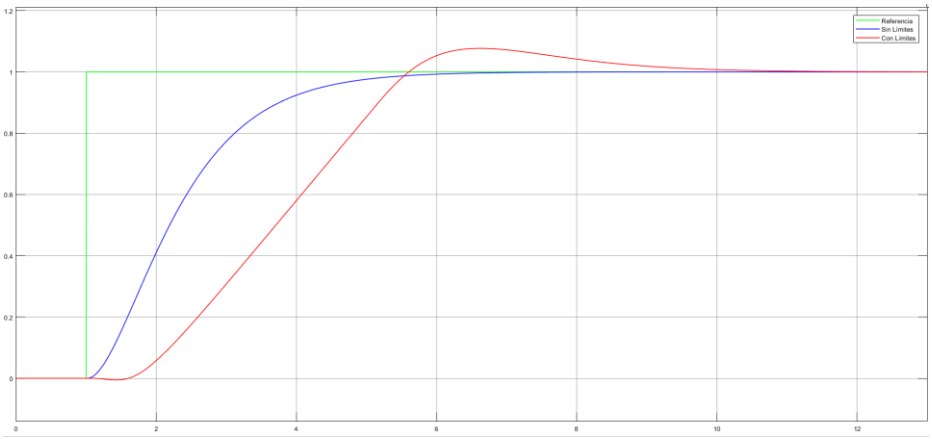

Figura 4-16. Resultados de la simulación para comparar efectos de los límites físicos

Tal y como se puede observar en los resultados de la simulación, la presencia de dichos límites físicos produce cambios en la respuesta del sistema, haciendo que este tenga un tiempo de establecimiento superior y presente sobreoscilación. Siendo

la nueva curva mucho más próxima a la respuesta real que se obtendrá del laboratorio remoto.

5. Laboratorio remoto.

En este capítulo se lleva a cabo el desarrollo de un laboratorio remoto para un sistema barra-bola. Para ello, a partir de la maqueta anteriormente descrita, se obtendrá el esquema visible en la Figura 5-1.

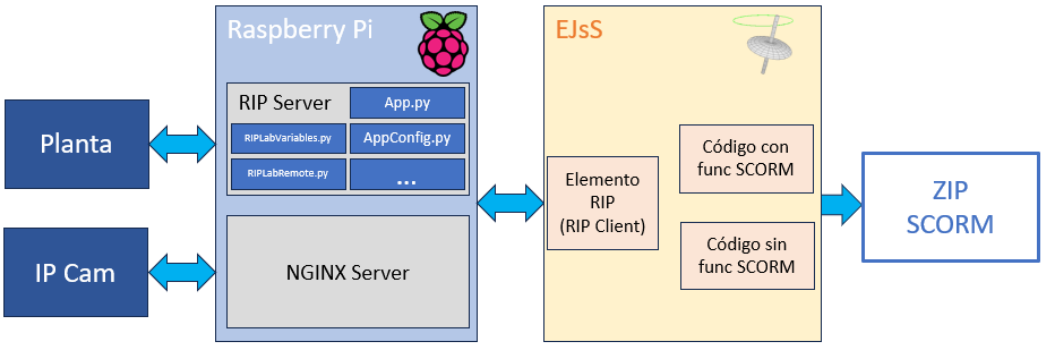

Figura 5-1. Esquema general del laboratorio remoto

- Creación de un servidor RIP (Remote Interoperability Protocol) el cual puede ser usado en una Raspberry Pi, permitiendo así interactuar con la planta real.
- Uso de una IP Cam para retransmitir en vivo todo lo que ocurre en la planta, pudiendo así visualizarla en tiempo real.
- Creación de un servidor NGINX en la Raspberry Pi, el cual es usado como proxy inverso, permitiendo así al cliente visualizar la imagen capturada por la IP Cam.
- Hacer uso de EJsS (Easy Java script Simulations) para crear la interfaz gráfica del cliente.
- Usar un archivo SCORM para la integración del laboratorio remoto en LMS (Learning Management System).

5.1. Servidor RIP.

Antes de pasar a la creación del servidor RIP, se debe de entender qué es el protocolo RIP, este consiste en un protocolo de comunicaciones que se basa en la conexión cliente-servidor. Ha sido diseñado especialmente para OLs (online laboratories) donde los clientes acceden desde un navegador web. RIP ofrece un mecanismo simple a sus usuarios para adquirir información sobre las experiencias del laboratorio y sobre las distintos inputs y outputs del laboratorio.

Las principales características que el protocolo ofrece son:

- Determinar las experiencias del laboratorio online.
- Obtención de meta-datos relacionados con cada experiencia.
- Obtención de una lista de variables de tipo lectura y escritura.
- Obtención de una serie de métodos para leer y escribir las variables.
- Capacidad de recurrir a métodos para leer y escribir las variables.
- Definir eventos del servidor para enviar datos de manera periódica.
- Suscribir un cliente a un evento declarado en una experiencia.

Es importante saber que el protocolo RIP tiene dos pilares, los métodos HTTP (POST y GET) para las comunicaciones y en el formato JSON-RPC, tal y como queda reflejado en la Figura 5-2.

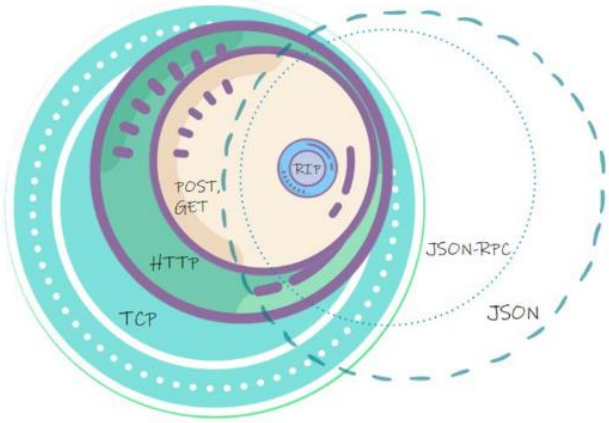

Figura 5-2. RIP se basa en los métodos HTTP y en el formato JSON-RPC

Por último, antes de pasar a la configuración del servidor RIP, es importante tener cuenta los distintos métodos de comunicación HTTP de los que dispone el protocolo RIP.

- GET: para obtener metadatos del laboratorio online.
- POST: para enviar solicitudes de cliente a servidor, escribiendo así los valores de las variables y leyendo los valores de las variables del laboratorio online.
- SSE: para suscribir a un cliente a flujos de datos, recibiendo así actualizaciones de los valores de las variables del laboratorio online.

5.1.1. Configuración del servidor RIP.

Una vez entendidos los principios básicos del protocolo RIP, se puede llevar a cabo la creación del servidor. Para ello, lo primero que se debe hacer, es descargar el software necesario, el cual se encuentra disponible en GitHub[1].

A continuación, se debe de crear un entorno virtual en el dispositivo en el que se va a desarrollar el servidor, en este caso en la Raspberry Pi. Para ello se debe introducir las siguientes instrucciones en la ventana de comandos:

pip3 install virtualenv

virtualenv -p python3 venv

Por otro lado, es indispensable la instalación de ciertas librerías para poder actuar sobre los pines de la Raspberry Pi y para poder trabajar con estructuras de tipo JSON, por lo que también será necesario introducir los siguientes comandos:

venv/bin/pip install cherrypie ujson RPi.GPIO

Una vez realizados los pasos anteriores, se está en disposición de modificar los distintos ficheros que permiten poner en marcha el servidor. La arquitectura del servidor queda reflejada en la Figura 5-3.

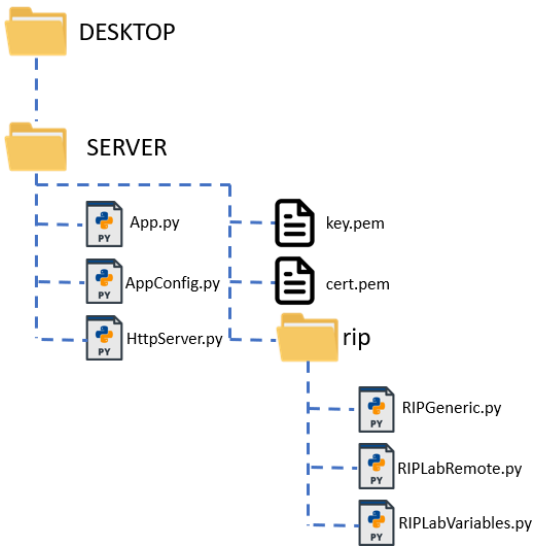

Figura 5-3. Arquitectura del servidor RIP

[1] Enlace para descarga de Servidor RIP

- *App.py*: es el fichero encargado de la inicialización del servidor, para ello lleva a cabo la importación de las clases necesarias para la configuración y funcionamiento del mismo.
- *AppConfig.py*: es el fichero encargado de la parametrización del servidor. En este, se fijan parámetros tan importantes como la IP y puerto del servidor además de si se desea usar seguridad, tal y como se puede observar en la Figura 5-4. Se debe de tener en cuenta, que para el uso que se le va a dar, es obligatorio implementar el servidor con seguridad, para que pueda ser integrado posteriormente en LMS. En este fichero también se indican las variables indicando su nombre, descripción, valor mínimo y máximo, precisión y se deben de clasificar como de lectura y/o escritura, tema que se tratará en el capítulo 5.1.2.

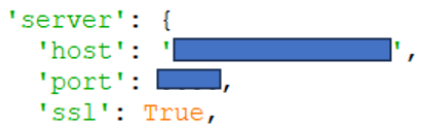

Figura 5-4. Parametrización del servidor en AppConfig.py

- *HttpServer.py*: este fichero tiene como finalidad activar el servidor HTTPS, para gestionar todos los mensajes recibidos (GET) y enviar respuestas (POST).
- *key.pem*: es una clave privada RSA que se genera junto al certificado *cert.pem* [19].
- *cert.pem*: es un documento de texto, el cual contiene un documento autofirmado para realizar la comunicación de forma segura mediante SSL. Para generar dicho certificado y la clave mencionada anteriormente, se pueden seguir las instrucciones especificadas en la parte final del fichero *HttpServer.py*.
- *RIPGeneric.py*: este fichero da soporte a la funcionalidad común del protocolo RIP. Los siguientes dos archivos que se van a mencionar, serán totalmente dependientes del caso de uso.
- *RIPLabVariables.py*: en este fichero se definen variables globales que serán usadas desde distintos ficheros, además de llevar a cabo la configuración de los pines de la Raspberry Pi.

- *RIPLabRemote.py*: en este fichero se recogen las funciones del laboratorio remoto, entre las que se pueden destacar la de lectura de pulsos del encoder, la que lleva a cabo el procesado de la imagen para obtener la posición de la bola y un bucle que llevará a cabo el control automático. Al igual que en el fichero *AppConfig.py* se deben de especificar las variables especificando los mismos parámetros.

Tras hacer las modificaciones necesarias en los ficheros anteriormente descritos el servidor puede ser iniciado, para ello, desde la ventana de comandos de la Raspberry se debe introducir la siguiente instrucción:

venv/bin/python3 App.py

El servidor RIP, ya está funcionando, para comprobarlo tan sólo se debe de introducir lo siguiente en el navegador:

https://localhost:port/RIP

Se debe de obtener una estructura JSON con la información del servidor, tal y como se puede observar en la Figura 5-5.

{"experiences":{"list":[{"id":"Laboratorio Remoto"}],"methods":[{"url":"[redacted]:[redacted]/RIP","type":"GET","description":"Retrieves information (variables and methods) of the experiences in the server","params":[{"name":"Accept","required":"no","location":"header","value":"application/json"},{"name":"expId","required":"no","location":"query","type":"string"}],"returns":"application/json","example":"http://[redacted]:[redacted]/RIP?expId=Laboratorio Remoto"}]}}

Figura 5-5. Respuesta JSON del servidor

5.1.2. Envío y recibo de variables.

En este apartado, se explican los cambios que se deben de hacer en los ficheros *AppConfig.py* y *RIPLabRemote.py* para poder realizar de manera correcta el envío y recibo de variables entre el servidor y el cliente.

Lo primero que se debe de tener claro, es que se va a trabajar con dos tipos de variables, tal y como ya se ha comentado anteriormente, las de escritura y las de lectura. Las de lectura, son aquellas que se envían desde el servidor hacia el cliente, mientras que las de escritura, son las que el servidor recibe desde el cliente. Teniendo esto claro, en la Figura 5-6 se pueden visualizar las modificaciones que se requieren en *AppConfig.py* y en *RIPLabRemote.py* para una variable de escritura (led) y otra de lectura (tiempo).

```
def default_info(self):
        '''
        You can provide default metadata here. AppConfig will override this definition.
        '''
        return{
            'name': 'Laboratorio Remoto',
            'description': 'Implementación del control en la Raspberry',
            'authors': 'Eduardo Lucena Alonso',
            'keywords': 'RASPBERRY PI',
            'readables': [{
                'name':'time',
                'description':'Tiempo del servidor RIP en segundos',
                'type':'float',
                'min':'0',
                'max':'Inf',
                'precision':'0'
              }
            ],
            'writables': [{
                'name':'led',
                'description':'Encendido y apagado del panel LED',
                'type':'boolean',
                'min':'false',
                'max':'true',
                'precision':'',
              }
            ]
            }
```

Figura 5-6. Declaración genérica de variables de tipo escritura y lectura

Para finalizar el envío de la variable de tipo lectura, se debe de crear una función en el fichero *RIPLabRemote.py*. Para el ejemplo anterior, esta función se puede visualizar en la Figura 5-7.

```
def getValuesToNotify(self):
    '''
    Variables to include in periodic SSE updates
    '''
    return[['time'],[self.sampler.lastTime()]
```

Figura 5-7. Función encargada del envío de flujo de datos

La solución por la que se ha optado para poder operar sobre la maqueta una vez recibida la variable de tipo lectura, se puede ver en la Figura 5-8.

```
#FUNCIONES CONTROL LED
@property
def led(self):
    return self._led

@led.setter
def led(self, value):
    self._led = value
    self.iluminacion(value)

def iluminacion(self, value):
    if value:
        GPIO.output(VAR.PIN_LED, GPIO.LOW)    #enciende panel led
    else:
        GPIO.output(VAR.PIN_LED, GPIO.HIGH)   #apaga panel led
```

Figura 5-8. Ejemplo de encendido y apagado del panel led

De esta manera, siguiendo las mismas estructuras de los ejemplos anteriores, se pueden recibir y enviar las variables que se deseen, además de actuar sobre la maqueta según los valores de las mismas. A continuación, en la Tabla 4 y en la Tabla 5 se facilitan todas las variables necesarias para el caso particular del control automático del sistema barra-bola.

Variable	Tipo	Función
time	float	Tiempo del servidor
dist-bola	float	Posición en cm de la bola
pmacabada	boolean	Booleano que indica que la puesta en marcha ha finalizado
fincontrol	boolean	Booleano que indica que el control automática ha finlaizado

Tabla 4. Variables de tipo lectura para control del sistema barra-bola

Variable	Tipo	Función
led	boolean	Permite apagar o encender el panel led
puestaenmarcha	boolean	Permite llevar a cabo una puesta en marcha, para comenzar en unas mismas condiciones iniciales
kp	float	Acción proporcional del controlador
ti	float	Tiempo integral del controlador
td	float	Tiempo derivativo del controlador
marchaparo	boolean	Booleano que permite iniciar el control automático
paremerg	boolean	Booleano que permite hacer una parada una vez ha comenzado a hacerse el control de la planta

Tabla 5. Variables de tipo escritura para control del sistema barra-bola

5.1.3. Funcionalidades principales del sistema barra-bola.

En este capítulo se explican las distintas funciones que se han definido en el fichero *RIPLabRemote.py* para el correcto funcionamiento del caso particular del sistema barra-bola. Estas funciones son las siguientes:

- Puesta en marcha: el objetivo de esta función es el de llevar a cabo una puesta en marcha, la cual permita realizar el control automático siempre desde unas mismas condiciones iniciales, además de inicializar las variables que sean necesarias. Para ello se mide el área que se forma en la parte inferior izquierda de la barra, la cual se puede observar en la

Figura 5-9 y se hace girar el motor en sentido antihorario hasta que se obtenga el valor de área deseado.

Figura 5-9. Área medida para puesta en marcha

- Posición de la bola: la finalidad de esta función es la de realizar el procesamiento de la imagen captada por la cámara para posteriormente obtener el centroide de la bola a partir de dicha imagen procesada. Lo primero que se debe de tener claro, es que la referencia que se ha tomado es la del eje de giro de la barra, ya que es un punto inmóvil. Teniendo este punto de referencia y el centroide de la barra, se está en disposición de obtener la distancia entre ambas usando la distancia euclídea [20], presente en la Ecuación 94.

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 - (y_2 - y_1)^2}$$

Ecuación 94

Sin embargo, la distancia que se necesita no es respecto al eje de giro, sino respecto al centro de la barra. Esto se puede se puede resolver usando trigonometría básica, debido a que se forma un triángulo rectángulo, justo como se puede observar en la Figura 5-10.

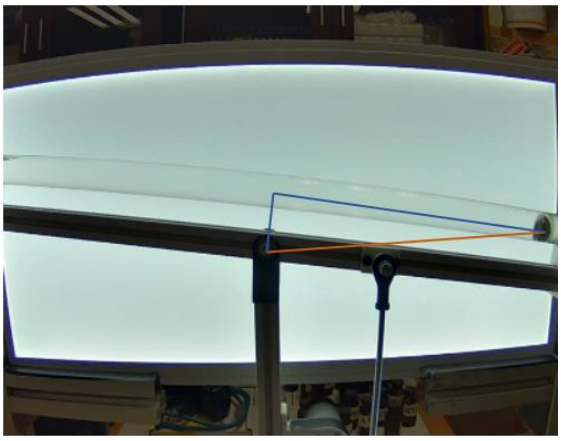

Figura 5-10. Trigonometría para obtener la distancia

El funcionamiento de dicho algoritmo viene detallado en el flujograma de la Figura 5-11. Por último, es importante mencionar que para el procesamiento de la imagen se ha hecho uso de la librería OpenCV.

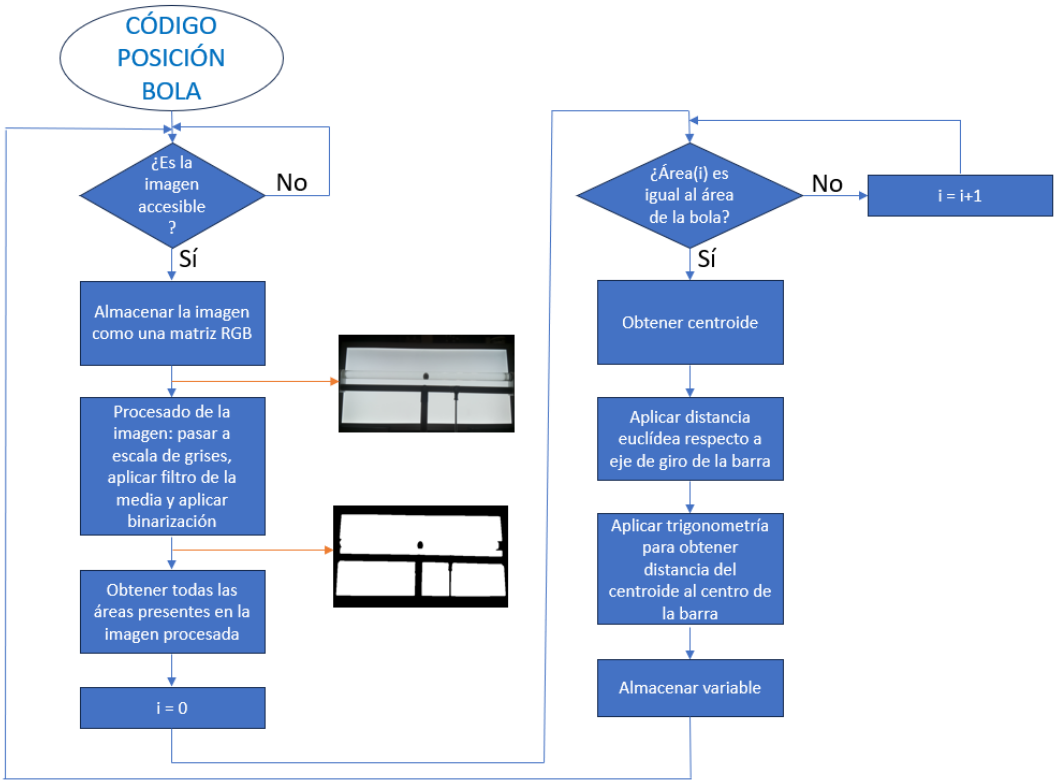

Figura 5-11. Flujograma de la función de la posición de la bola

- Lectura de pulsos del encoder: la finalidad de este algoritmo es, obtener a partir de las señales cuadradas que emite el encoder del motor, los pulsos del mismo, llegando así a saber la posición angular y el sentido de giro del motor.

Para ello y sabiendo cómo funcionan las salidas del encoder, que ya fue desarrollado en el apartado 3.1.1, este algoritmo almacena el estado de las salidas del encoder. Las distintas combinaciones se encuentran recogidas en la Tabla 6.

Canal encoder A	0	0	1	1
Canal encoder B	0	1	0	1

Tabla 6. Posibles combinaciones de las salidas del encoder

De esta manera, comprobando el estado actual de las salidas del encoder y el estado anterior, se puede saber en qué sentido está girando el motor y contabilizar los pulsos del mismo para saber en qué posición se encuentra, tal y cómo se puede observar en la Figura 5-12.

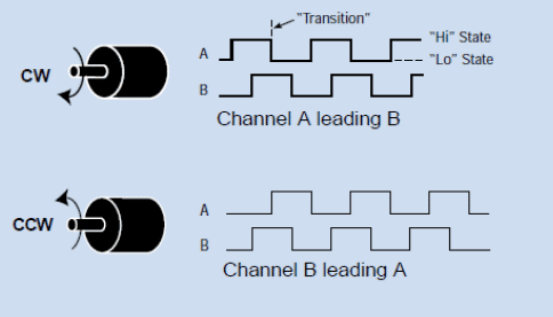

Figura 5-12. Sentido de giro del motor en función de los canales del encoder

- Control: este algoritmo tiene la finalidad de llevar a cabo el control del sistema barra-bola siguiendo el lazo de control que se desarrolló en el capítulo 4.2. Para ello, lo primero que debe de tenerse en cuenta, es que el control se hace de manera discreta, por lo que para aplicar la acción derivativa que se requiere, se deberá de hacer la siguiente aproximación:

$$\frac{de(t)}{dt} \approx \frac{e - e_{prev}}{t - t_{prev}}$$

Ecuación 95

Una vez entendido esto, y recordando que se trata de un control en cascada, se puede pasar al funcionamiento del algoritmo, el cual está presente en el flujograma de la Figura 5-13.

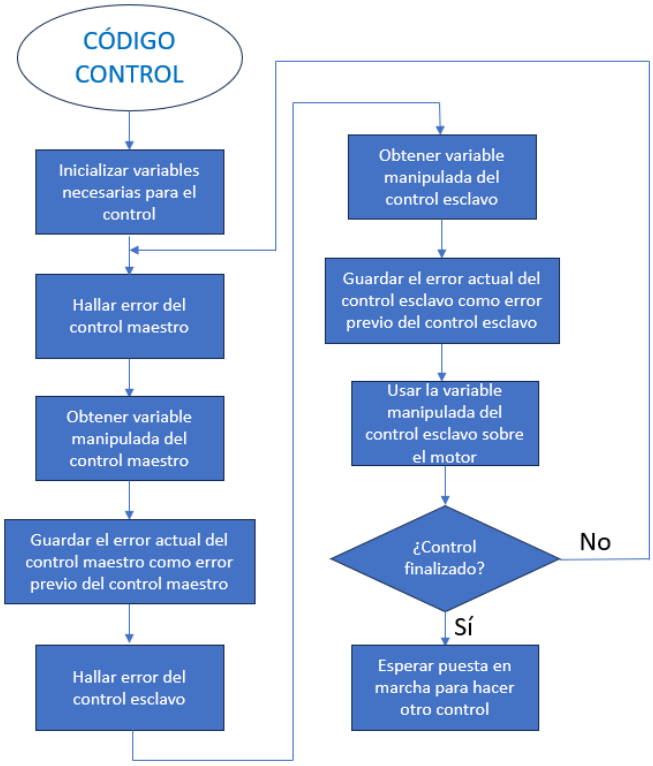

Figura 5-13. Flujograma del algoritmo de control

5.1.4. Reinicio automático del servidor

A continuación, se llevará a cabo la creación de un archivo [21], que en caso de que se produjese un corte eléctrico o similar, será ejecutado justo al reiniciarse la Raspberry, permitiendo así volver a activar el servidor.

Para ello, será necesario agregar un comando al archivo /etc/rc.local que se ejecuta al iniciar el sistema, el cual se lanzará desde un terminal con la siguiente línea:

sudo nano /etc/rc.local

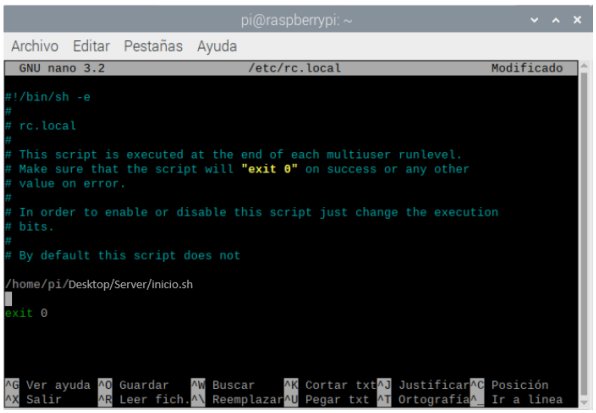

Figura 5-14. Archivo /etc/rc.local

En la siguiente figura, se muestran las instrucciones necesarias para la ejecución del servidor:

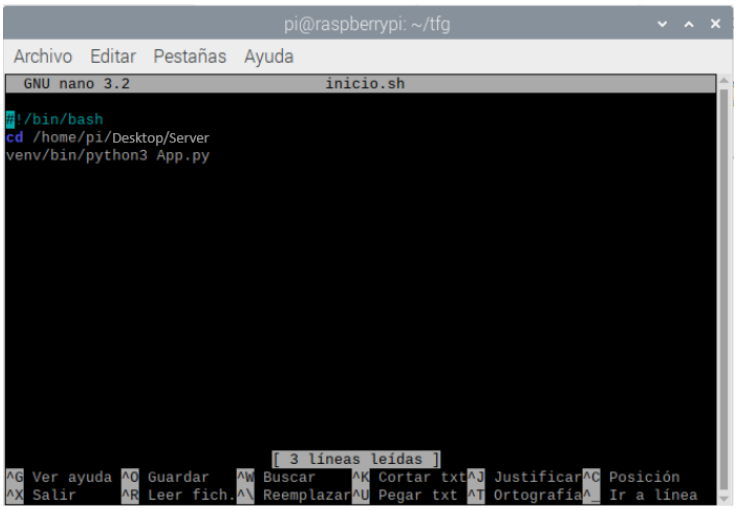

Figura 5-15. Archivo inicio.sh

Para finalizar hay que dar permisos de ejecución al fichero que se desea ejecutar, para ello, en la ventana de comandos se introducirá la siguiente instrucción:

sudo chmod +x inicio.sh

De esta manera, se puede garantizar, que el servidor siempre estará activado.

5.2. IP Cam.

Una vez creado el servidor RIP, el siguiente paso es usar una cámara que permita retransmitir la imagen de todo lo que ocurre en el laboratorio remoto, para que así el cliente pueda visualizarlo en tiempo real. Para ello se deberá de llevar a cabo la configuración de la cámara y la posterior creación de un servidor NGINX.

5.2.1. Configuración de la IP Cam.

La cámara usada para el proyecto se trata de una AXIS M3044-V [22], esta es una cámara minidomo, fija, con calidad HDTV a 720p. Dicha cámara se puede visualizar en la Figura 5-16.

Figura 5-16. Cámara IP usada en el laboratorio remoto

Para la configuración de dicha cámara lo primero que se debe de hacer es localizar el dispositivo en la red y asignarle una dirección IP, para ello se debe utilizar AXIS IP Utility o AXIS Device Manager. Ambas aplicaciones son gratuitas y pueden descargarse desde la página web de AXIS[2].

A continuación, se debe acceder desde el navegador a dicha dirección. Si es la primera vez que se accede al dispositivo, se deberá de establecer la contraseña root. Una vez se haya accedido se podrá visualizar la interfaz de configuración de la cámara, la cual está presenta en la Figura 5-17.

Figura 5-17. Interfaz de configuración de la IP Cam

[2] Aplicaciones para obtener dirección de IP Cam

Siendo:

1. Barra de control de visualización en directo.
2. Visualización en directo.
3. Nombre de producto.
4. Información del usuario, temas de colores y ayuda.
5. Barra de control de vídeo.
6. Conmutador de ajustes.

Para el caso particular del laboratorio remoto la única configuración que se debe llevar a cabo, se realiza desplegando el conmutador de ajustes de la Figura 5-17 y accediendo a la pestaña de “Sistema”, visible en la Figura 5-18.

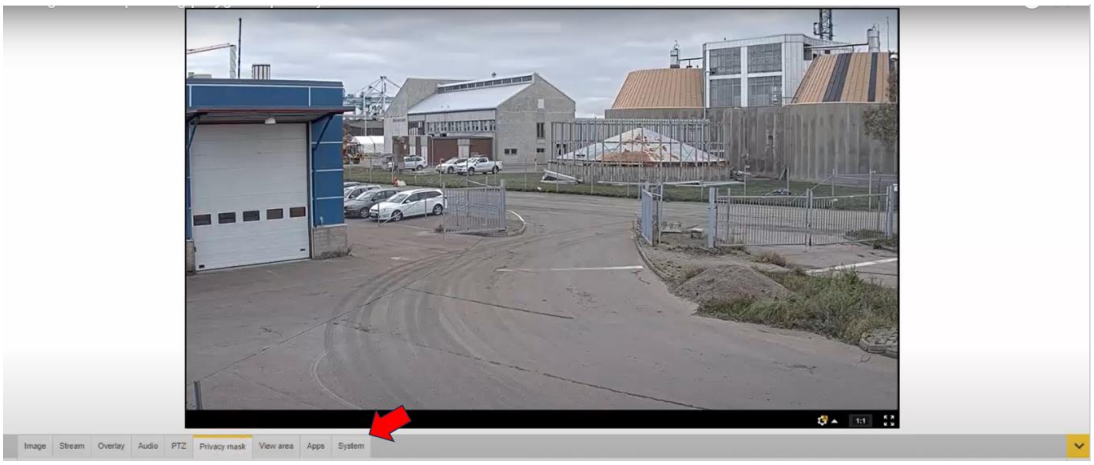

Figura 5-18. Pestaña para configuración de la IP Cam

Una vez dentro de dicha pestaña, se debe activar el protocolo http y desactivar el https, esto se hace para el posterior uso del servidor NGINX.

Esta simple configuración será más que suficiente para el uso de la IP Cam en el laboratorio remoto, aun así, si se desea llevar a cabo una configuración más compleja porque se requiera de otras funcionalidades que tiene la cámara, se deberá de acceder al manual de usuario de dicha cámara[3].

[3] Manual de Usuario IP Cam

5.2.2. Creación del servidor NGINX.

Lo primero que hay que dejar claro es el por qué se decide usar un servidor NGINX, ya que tras la configuración de la IP Cam sería lógico pensar que el cliente podría acceder a la dirección de dicha cámara. Sin embargo, para evitar posibles problemas de autenticación, se ha optado por hacer uso de un servidor NGINX [23].

El funcionamiento base de NGINX es similar al de otros servidores web, en el que un usuario realiza una petición a través del navegador al servidor, y este le envía la información solicitada al navegador. Lo que hace diferente a NGINX es su arquitectura a la hora de manejar procesos, ya que otros servidores web como Apache crean un hilo por cada solicitud.

Tras entender por qué se usa dicho servidor y su funcionamiento básico, el siguiente paso es explicar cómo se descarga en la Raspberry. Para ello, en una terminal, se seguirán los siguientes pasos:

- Instalar NGINX:

sudo apt-get install nginx

- Iniciar servidor:

sudo /etc/init.d/nginx start

- Comprobar desde la Raspberry que el servidor está iniciado, para ello basta con acceder a http://localhost.

Antes de explicar la configuración de dicho servidor para que permita al cliente visualizar la imagen de la IP Cam, se deben generar los certificados autofirmados del servidor NGINX, para así poder usar el protocolo https, lo cual permitirá la integración del laboratorio remoto en LMS.

Para generar dicho certificado autofirmado, desde la terminal de Raspberry se introduce la siguiente instrucción:

sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/nginx.key -out /etc/ssl/certs/nginx.crt

Tras introducir dicha instrucción, se pedirá información sobre la organización que está desarrollando el servidor, tal y como se puede observar en la Figura 5-19. Lo único que realmente es importante es el “Nombre común”, donde debe establecerse la dirección IP del servidor.

```
Country Name (2 letter code) []:
State or Province Name (full name) []:
Locality Name (eg, city) []:
Organization Name (eg, company) []:
Organizational Unit Name (eg, section) []:
Common Name (eg, fully qualified host name) []: your_ip_address
Email Address []:
```

Figura 5-19. Información requerida para servidor NGINX

En caso de que se requiera algún tipo de configuración adicional se recomienda acceder a la siguiente dirección dónde se desarrolla con mayor detenimiento el uso de dicho certificado autofirmado[4].

A continuación, se detalla la configuración particular que se ha realizado del servidor NGINX para su uso en el laboratorio remoto.

Para realizar las modificaciones necesarias, primero se debe de introducir la siguiente instrucción en la terminal de la Raspberry.

cd /etc/nginx/sites-enabled/default

Para el caso particular del laboratorio remoto del sistema barra-bola, dicho archivo ha sido modificado, el resultado se puede visualizar en la Figura 5-20.

[4] Configuración de certificado autofirmado de servidor NGINX.

```
Server{
    listen 443 ssl;

    # RSA certificate
    ssl_certificate /etc/ssl/private/cer.pem;
    ssl_certificate_key /etc/ssl/private/key.pem;

    # Redirect non-https traffic to https
    if ($scheme != "https"){
        return 301 https://$host$request_uri;
        }# managed by Certbot
    }

    location /cam {
        proxy_pass http://  IP de la IP CAM  /mjpg/1/video.mjpg;
        proxy_hide_header Access-Control-Allow-Origin;
        add_header Access-Control-Allow-Origin *;
        proxy_set_header Authorizacion "  Clave de acceso a IP Cam en base 64  ";
        proxy_pass_header Authorization;
        }
    }
```

Figura 5-20. Configuración de servidor NGINX

De esta manera, cuando el usuario acceda a la dirección IP de la Raspberry con la ruta /cam y protocolo https, el servidor NGINX hará de puente entre el cliente y la dirección de la IP Cam, haciendo posible visualizar la imagen desde la interfaz del usuario.

5.3. Cliente RIP.

El siguiente paso para el desarrollo del laboratorio remoto es la creación de la interfaz gráfica del usuario. Para el desarrollo de dicha interfaz se ha usado EJsS (Easy JavaScript Simulations). Esta es una herramienta de software diseñada para la creación de simulaciones informáticas discretas, pero debido a sus altas prestaciones, es una opción idónea para llevar a cabo la interfaz gráfica del usuario.

Debido al altísimo número de aplicaciones que se le puede dar a esta herramienta y a su cierta complejidad, se recomienda acceder al siguiente manual[5], donde se explica en profundidad cómo usar dicha herramienta. Este capítulo se centrará en cómo configurar EJsS para poder llevar a cabo las comunicaciones con el servidor RIP creado en el apartado anterior.

[5] Manual de EJsS

5.3.1. Conexión entre servidor y cliente.

Lo primero que se debe de realizar es la conexión entre servidor y cliente, para ello, una vez dentro de EJsS, se debe de agregar el elemento RIP, esto se hace en la pestaña de Modelo>Elementos>SoftwareLinks, tal y como se puede observar en la Figura 5-21.

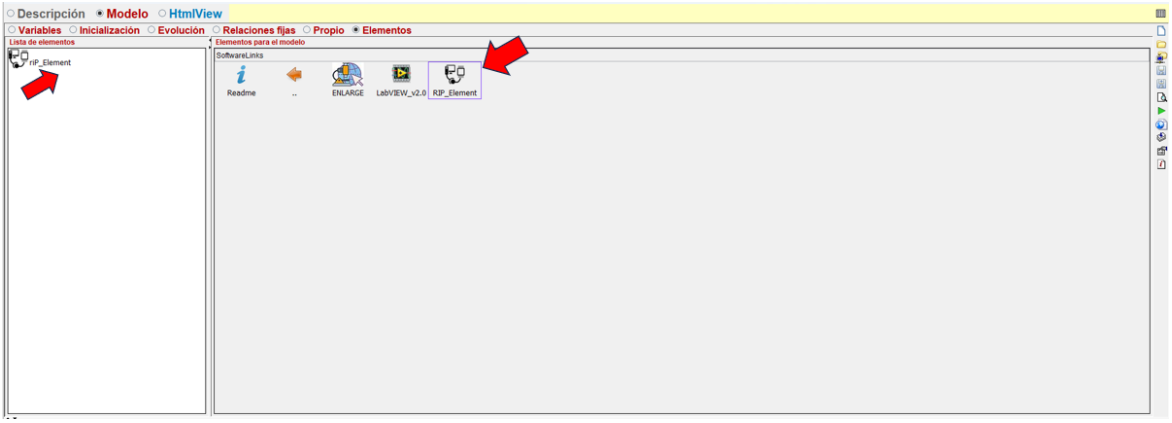

Figura 5-21. Elemento RIP necesario para la conexión

A continuación, se realiza la configuración del elemento RIP, este consta de tres distintas pestañas, tal y como se observa en la Figura 5-22.

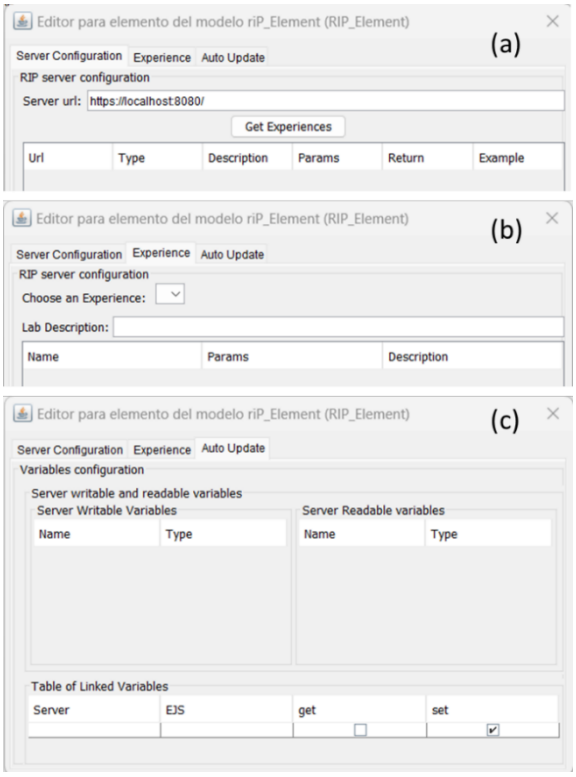

Figura 5-22. Configuración del elemento RIP. (a) Configuración de servidor. (b) Experiencias del servidor. (c) Pestaña de actualización automática

En la Figura 5-22 (a), se observa la pestaña donde se fija la dirección URL del servidor, que es la misma que la de la Raspberry. Pulsando *Get Experiences*, se obtiene información del servidor. En esta ventana, tras introducir la dirección URL del servidor y obtener las experiencias, los campos inferiores se completarán con la misma información que se obtiene al abrir el enlace en el buscador, es decir, tal y como se vio en la Figura 5-5.

En la Figura 5-22 (b), se visualiza la descripción del laboratorio que se haya modificado en los archivos AppConfig.py y RipLabRemote.py. En el campo inferior se observarán tres métodos de comunicaciones distintos.

En la Figura 5-22 (c), se obtienen todas las variables de tipo lectura y escritura que previamente se han definido en RipLabRemote.py y AppConfig.py. Desde esa misma pestaña en “Table of Linked Variables” se han de poner las variables, con el nombre que tienen el servidor y con el que tendrán en EJsS, por comodidad se recomienda utilizar los mismos, aunque no es obligatorio. Además, se deben de marcar las casillas get o set, si son de escritura o de lectura respectivamente.

De esta manera, y siguiendo con el ejemplo genérico realizado en el capítulo 5.1.2, las variables que se deben de agregar en esta tercera pestaña en el apartado de “Table of Linked Variables” se pueden visualizar en la Figura 5-23.

Table of Linked Variables

Server	EJS	get	set
led	led	☐	☑
time	time	☑	☐

Figura 5-23. Variables de tipo lectura y escritura en EJsS para ejemplo genérico

El siguiente paso, sería declarar dichas variables en la pestaña Modelo>Variables de EJsS, tal y como se puede ver en la Figura 5-24.

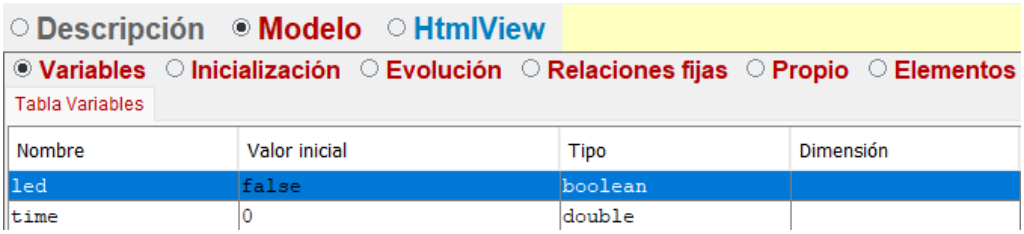

Figura 5-24. Declaración de variables en EJsS para ejemplo genérico

A continuación, se agrega la instrucción *rip.connect()* en la pestaña Modelo>Inicialización, de esta manera, la conexión entre servidor y cliente se llevará a cabo siempre que el cliente acceda a la interfaz de usuario.

El último paso es la creación de la interfaz gráfica desde HtmlView, para el caso de las variables genéricas se podría crear un botón de dos estados, el cual permitiría encender y apagar el panel de iluminación, un campo numérico, el cual permita al usuario ver cuánto tiempo lleva el servidor funcionando y un tercer elemento que permita visualizar la imagen de la IP Cam.

La pestaña HtmlView resultante es visible en la Figura 5-25.

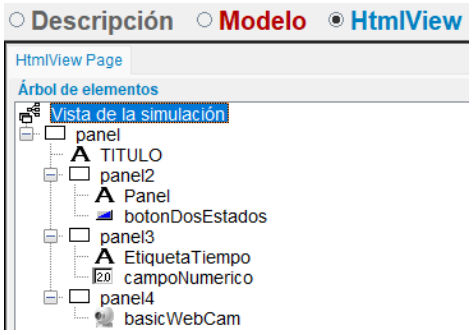

Figura 5-25. HtmlView Genérico

La configuración del botón de dos estados es la siguiente:

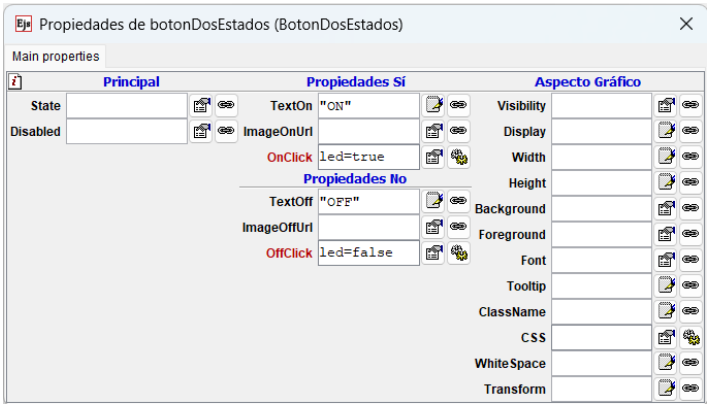

Figura 5-26. Configuración básica de botón de dos estados

La configuración del campo numérico se puede ver en la siguiente figura:

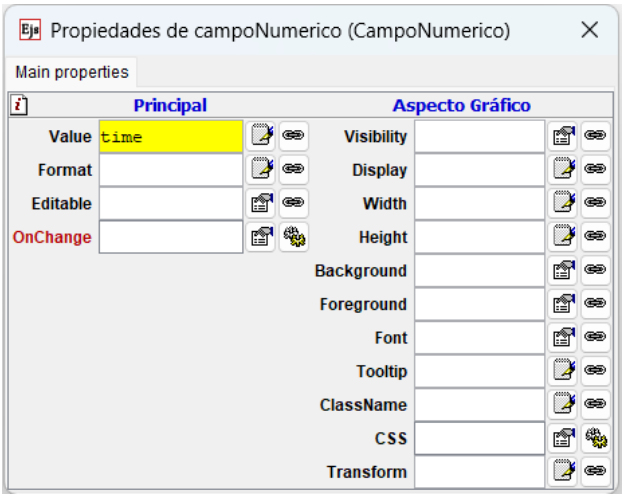

Figura 5-27. Configuración básica de campo numérico

Y por el último, la configuración del elemento llamado “basicWebCam”, es el siguiente:

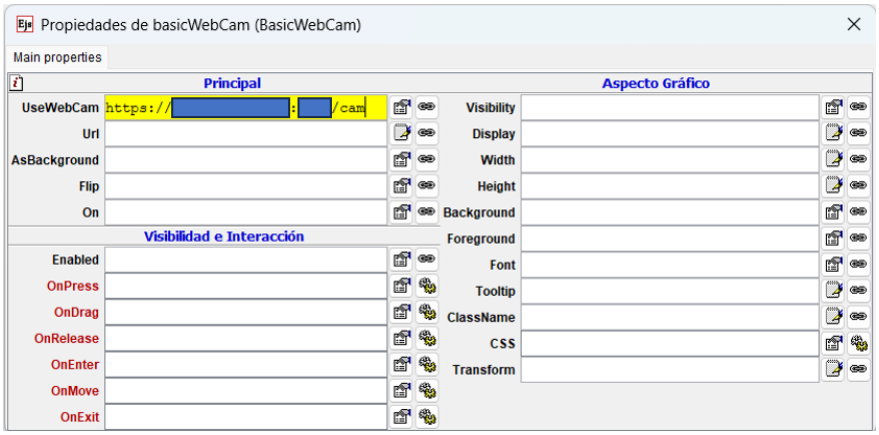

Figura 5-28. Configuración basicWebCam

El resultado de la interfaz de usuario que se ha desarrollado es visible en la Figura 5-29. Este permite activar y desactivar el panel de iluminación y visualizar el tiempo que lleva el servidor activo.

Figura 5-29. Resultado de la interfaz de ejemplo genérico

Aplicando esta misma estructura, se puede desarrollar el caso particular del sistema barra-bola. De esta manera, el apartado de Table of Linked Variables, para el caso particular se puede observar en la Figura 5-30.

Table of Linked Variables

Server	EJS	get	set
led	led	☐	☑
time	time	☑	☐
dist-bola	distancia	☑	☐
puestaenmarcha	puestaenmarcha	☐	☑
pmacabada	visibilidad	☑	☐
modotrabajo	modo	☐	☑
kp	kp	☐	☑
ki	ki	☐	☑
kd	kd	☐	☑
marchaparo	marchaparo	☐	☑
resetinicial	resetinicial	☐	☑
kiflag	kiflag	☑	☐
fincontrol	fincontrol	☑	☐
paremerg	emergencia	☐	☑

Figura 5-30. Variables de tipo lectura y escritura en EJsS para control del barra-bola

Y la pestaña de la interfaz gráfica resultante es visible en la Figura 5-31.

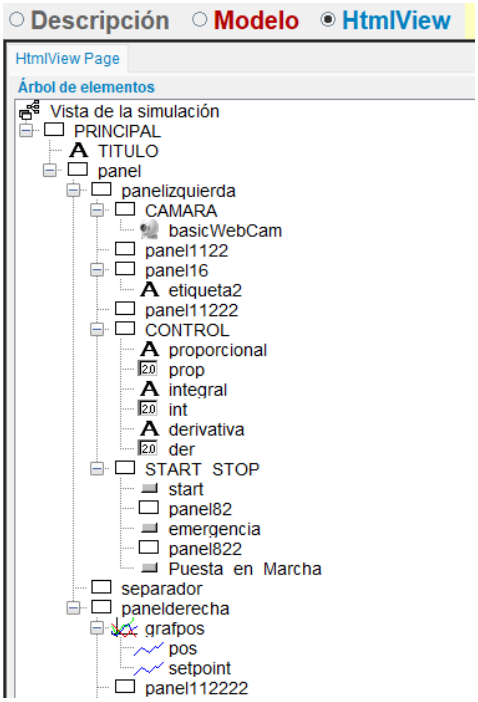

Figura 5-31. HtmlView para sistema barra-bola

5.4. Integración del laboratorio remoto con LMS mediante SCORM.

Para la integración del laboratorio remoto con LMS, se hará uso de distintas funciones, entre las que podemos destacar:

- scorm.getVersion(): permite obtener la versión del archivo.
- scorm.getLearnerName(): permite obtener nombre y apellidos del usuario que está utilizando la simulación.
- scorm.getLearnerId(): permite obtener el Id de ILIAS del usuario que está utilizando la simulación.
- scorm.setScoreMax(): permite establecer la puntuación máxima.
- scorm.setScoreMin(): permite establecer la puntuación mínima.

En el caso del ejemplo anteriormente descrito, se han usado en la pestaña Modelo>Inicialización, pudiendo visualizarse en la Figura 5-32.

```
version=_scorm.getVersion();
learnerName=_scorm.getLearnerName();
learnerId=_scorm.getLearnerId();
_scorm.setScoreMax("100");
_scorm.setScoreMin("0");
```

Figura 5-32. Uso de funciones para integración en LMS

Además, en la pestaña HtmlView, se ha agregado otro panel, en el cual se visualizará los datos del usuario, tal y como se puede observar en la Figura 5-33.

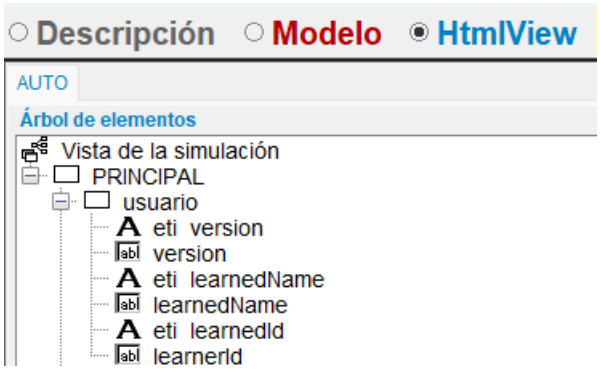

Figura 5-33. Panel de identificación de usuario

Para crear el paquete SCORM [24] y poder integrarlo en LMS, se debe de hacer click derecho en el panel derecho de EJsS, justo como viene indicado en la Figura 5-34, pulsando sobre la opción de crear el paquete SCORM.

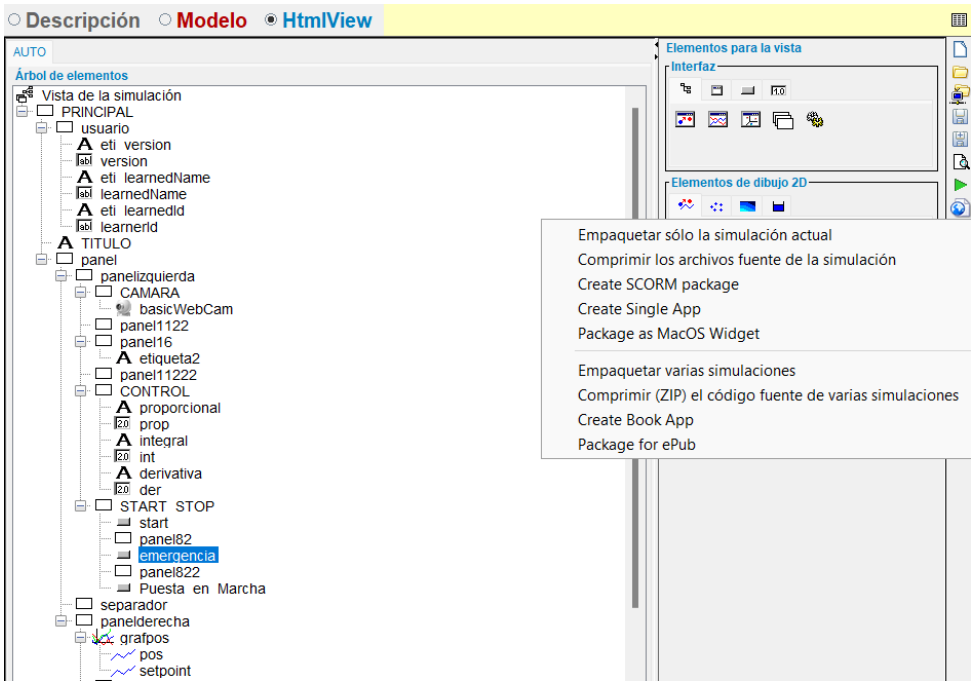

Figura 5-34. Creación del paquete SCORM

Tras esto, se debe de elegir la versión “2004” y desmarcar la casilla para no incluir la descripción, justo como aparece en la Figura 5-35.

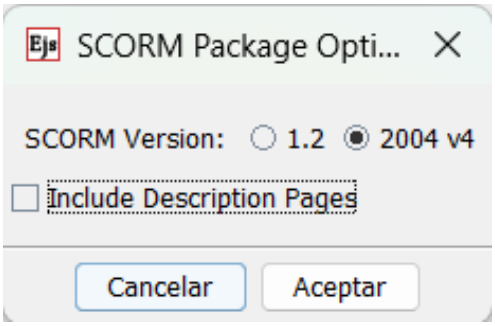

Figura 5-35. Configuración del paquete SCORM

Por último, este archivo con formato zip debe de ser integrado en LMS, para ello se debe crear una actividad, tal y como se puede observar en la Figura 5-36.

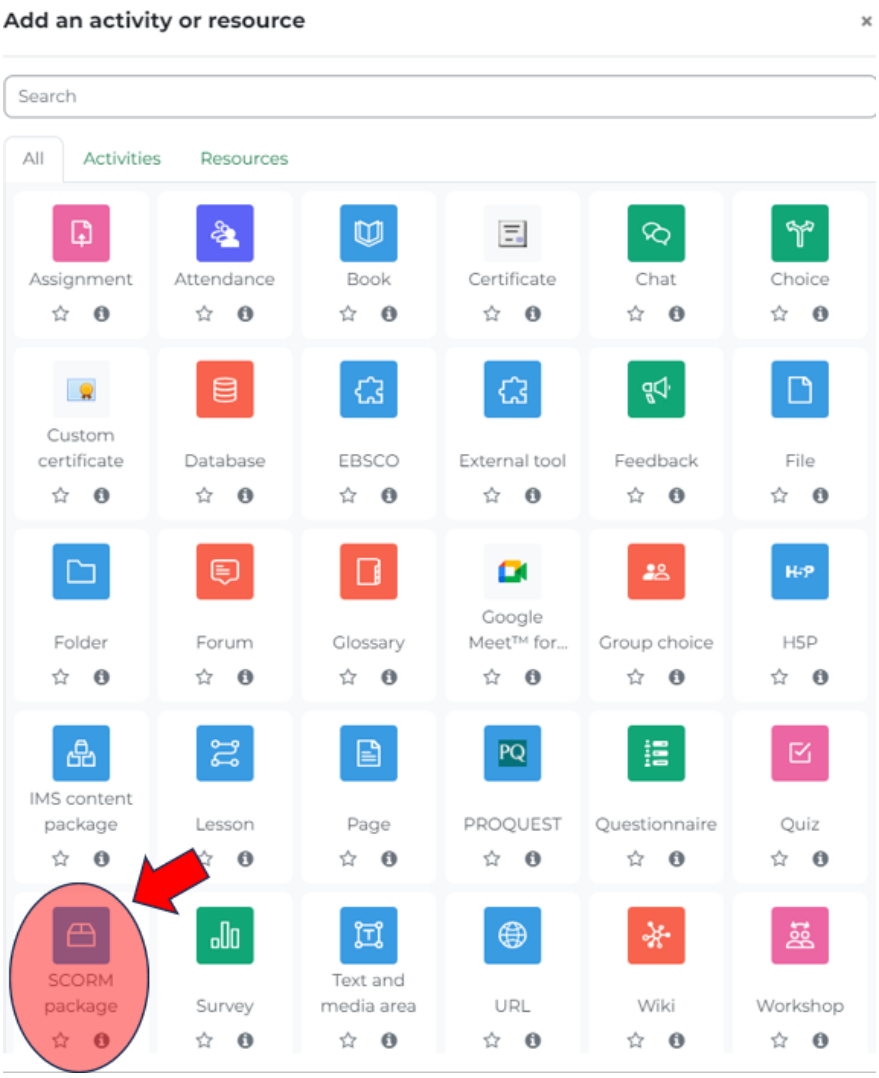

Figura 5-36. Creación de la actividad

El resultado del laboratorio remoto se puede visualizar en la Figura 5-37.

Figura 5-37. Laboratorio remoto integrado en LMS

De esta manera, el laboratorio remoto estaría terminado, pero solo sería accesible si el usuario se encuentra conectado a la misma red que la Raspberry. Para conseguir que sea accesible desde cualquier red, se debe realizar la conexión del siguiente apartado.

5.5. Esquema de conexiones del laboratorio remoto.

Como recién se ha comentado, para poder hacer que el laboratorio remoto sea accesible desde cualquier red, se usará un Switch Nortel 5510-24T y el Gateway de la UJA. Justamente como se ve reflejado en la Figura 5-38.

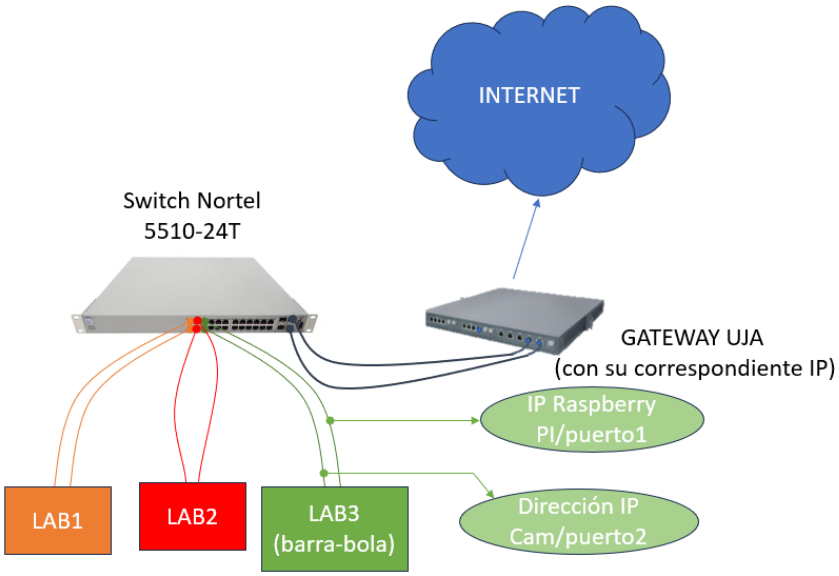

Figura 5-38. Esquema de conexiones del laboratorio remoto

En dicha figura, se puede visualizar como el switch, gracias a sus distintas tomas de Ethernet, permite conectar distintos laboratorios remotos. Gracias al GATEWAY de la UJA, se podrá acceder al servidor RIP y a la imagen de la IP Cam, para ello bastará con usar la IP del GATEWAY de la UJA especificando el puerto deseado, de esta manera, se hace la redirección, siendo así el laboratorio remoto accesible desde cualquier red.

5.6. Presupuesto de ejecución material.

A continuación, para finalizar este capítulo, en la Tabla 7, se adjunta un listado de los distintos componentes mencionados en el capítulo 3 y la IP Cam con sus respectivos precios, para así poder conocer el presupuesto de ejecución material de este laboratorio remoto.

Es importante tener en cuenta, que no se ha agregado el precio de mano de obra, ya que la realización del proyecto se ha desarrollado en todo momento como parte de un trabajo fin de grado y una parte considerable del tiempo invertido en su desarrollo ha sido el estudio de las tecnologías que se requerían para la creación de dicho laboratorio.

ELEMENTO	PRECIO (€)
Motor CC	51,98
Raspberry Pi 4 Model B	66,90
Cámara Raspberry v2.1	32,70
Panel de iluminación	44,99
Puente H	15,50
Relé	2,95
Adaptardor AC	21,99
IP Cam	232,95
Total	**469,96**

Tabla 7. Presupuesto de ejecución material

6. Laboratorio virtual.

En este capítulo, se lleva a cabo el desarrollo de un laboratorio virtual, basado en el sistema barra-bola descrito en los capítulos anteriores y su posterior integración en LMS. Para su desarrollo se hará uso de EJsS [25], debido al gran potencial que este ofrece. En la Figura 6-1, se puede observar el esquema del funcionamiento básico de dicho laboratorio virtual.

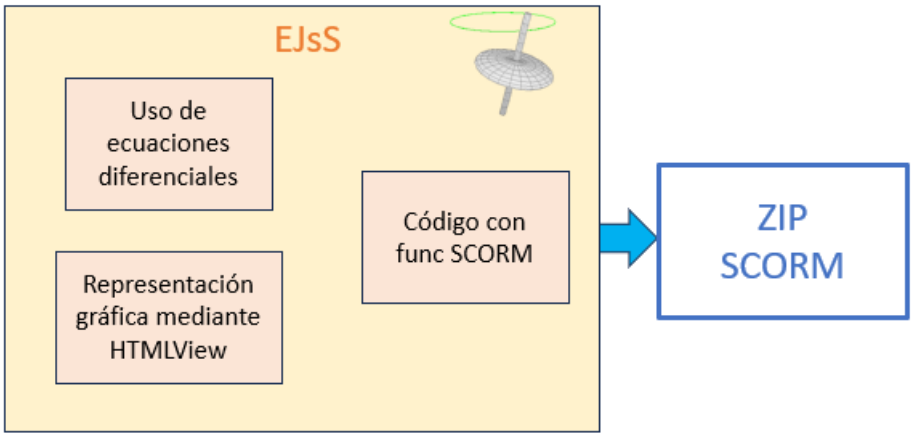

Figura 6-1. Esquema general del laboratorio virtual

Los motivos por los cuales se ha elegido EJsS para llevar a cabo este laboratorio virtual son los siguientes:

- Permite el uso de ecuaciones diferenciales, lo cual hará posible estudiar la variación de ciertas variables en función del tiempo basándose en las ecuaciones diferenciales obtenidas en el capítulo 2.
- Permite representar gráficamente el sistema barra-bola, emulando así el comportamiento de la maqueta real.
- Permite la integración en LMS mediante la generación de un archivo zip, pudiendo ser usado así en docencia. La integración en LMS no se desarrollará en este capítulo, para ello se debe acudir al capítulo 5.4, ya que se trata del mismo proceso.

Antes de comenzar con el desarrollo del laboratorio virtual, se debe tener claro el lazo de control de la Figura 4-7 y la nomenclatura usada.

6.1. Variables usadas para el desarrollo del laboratorio virtual.

Para el correcto desarrollo de este laboratorio virtual se han usado una serie de variables, las cuales, por motivos de organización, se han clasificado en distintas pestañas dentro de EJsS. Dichas pestañas, son las siguientes:

- “CONSTANTES”: en esta se pueden encontrar la longitud del brazo del motor, la longitud de la barra, la masa de la bola, la gravedad, la viscosidad del fluido, el radio de la bola y por último una variable llamada “escala”, que relaciona el tamaño de los elementos que se usarán para crear la interfaz gráfica con el tamaño de los elementos reales, posteriormente se mostrará su uso.
- “VARIABLES MOTOR”: en esta se pueden encontrar el ángulo de giro del motor en radianes, el ángulo de giro del motor en revoluciones, la velocidad angular del motor en rps y la ganancia del motor y constante de tiempo del motor, que permitirán definir la dinámica del motor.
- “VARIABLES BARRABOLA”: en esta se encuentran el ángulo de giro de la barra, la posición de la bola y la velocidad de la misma.
- “VARIABLES CONTMOTOR”: en esta pestaña se encuentran las variables involucradas en el control esclavo desarrollado en el capítulo 4.3.1. Siendo estas variables la acción proporcional, la acción derivativa, el error actual del motor, la aproximación de la derivada del error, la diferencia de tiempo entre la toma del error actual y la toma del error anterior, el error anterior, el tiempo cuando se ha tomado el error anterior y la salida del controlador esclavo.

  Es importante tener en cuenta que la acción proporcional y la acción derivativa ya vienen definidas, ya que este laboratorio virtual está diseñado para que el usuario sólo pueda interactuar con el control maestro. Por otro lado, este control esclavo no tiene acción integral ya que cómo se explicó en el capítulo 4, el uso de esta acción produce inestabilidad.

- “VARIABLES CONTBARBOL”: en esta pestaña se encuentran las variables involucradas en el control maestro desarrollado en el capítulo 4.3.3. Siendo estas variables la acción proporcional, la constante de tiempo de la acción integral, la constante de tiempo de la acción derivativa, el error actual de la posición de la bola, la aproximación de la integral del error, la aproximación de la derivada del error, la diferencia de tiempo entre la toma del error actual y la toma del error anterior, el error anterior, el tiempo cuando se ha tomado el error anterior, la salida del controlador maestro que será la referencia para el lazo del control esclavo y la referencia de la posición de la bola.

  Es importante observar que la acción proporcional, y las constantes de tiempo, derivativa e integral, en este caso tendrán valores nulos, ya que será el usuario quien introduzca los valores que considere oportunos para ver así cómo responde el sistema.

- “VARIABLES TEMPORALES”: donde se encuentran las variables que se usarán para la implementación las ecuaciones diferenciales.

Una vez se tienen claras las variables que se van a usar, y cómo se han organizado, pueden introducirse en EJsS, siendo el resultado el mismo que se visualiza en la Figura 6-2.

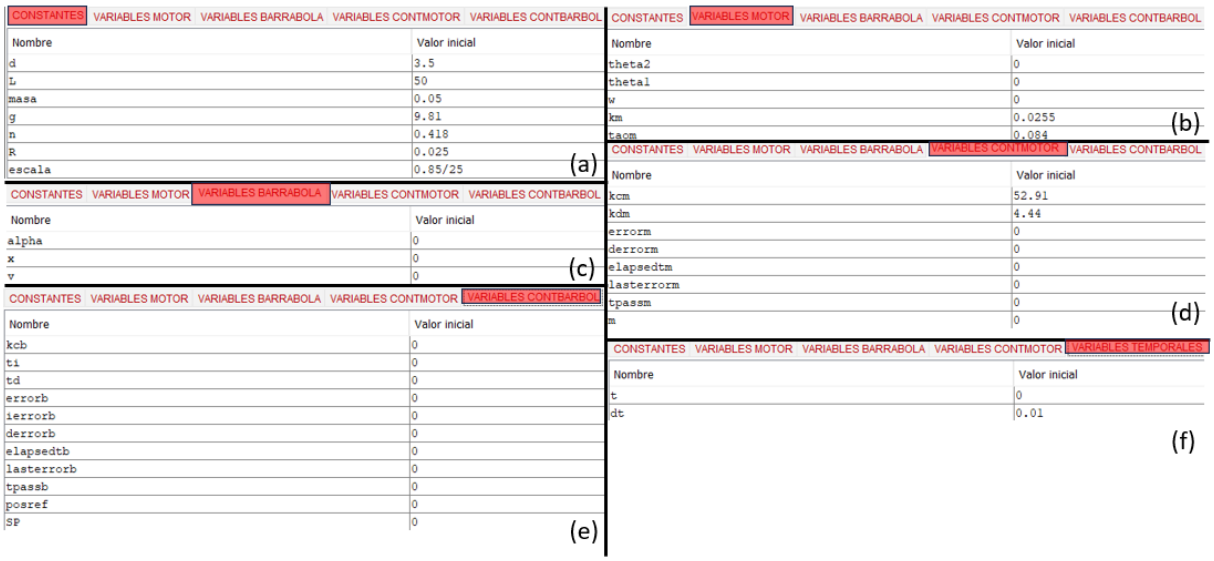

Figura 6-2. Declaración de variables en EJsS para laboratorio virtual. (a) CONSTANTES. (b) VARIABLES MOTOR. (c) VARIABLES BARRABOLA. (d) VARIABLES CONTMOTOR. (e) VARIABLES CONTBARBOL. (f) VARIABLES TEMPORALES.

6.2. Evolución de variables.

Una vez se conocen las variables que se usan para el desarrollo de este laboratorio virtual, el siguiente paso es conseguir que la simulación tenga un comportamiento lo más parecido a la realidad, para ello se hará uso de dos herramientas que EJsS facilita, la primera permite la implementación de ecuaciones diferenciales y la segunda permite usar código. Dichas herramientas se encuentran en la pestaña de Modelo>Evolución, tal y como se puede observar en la Figura 6-3.

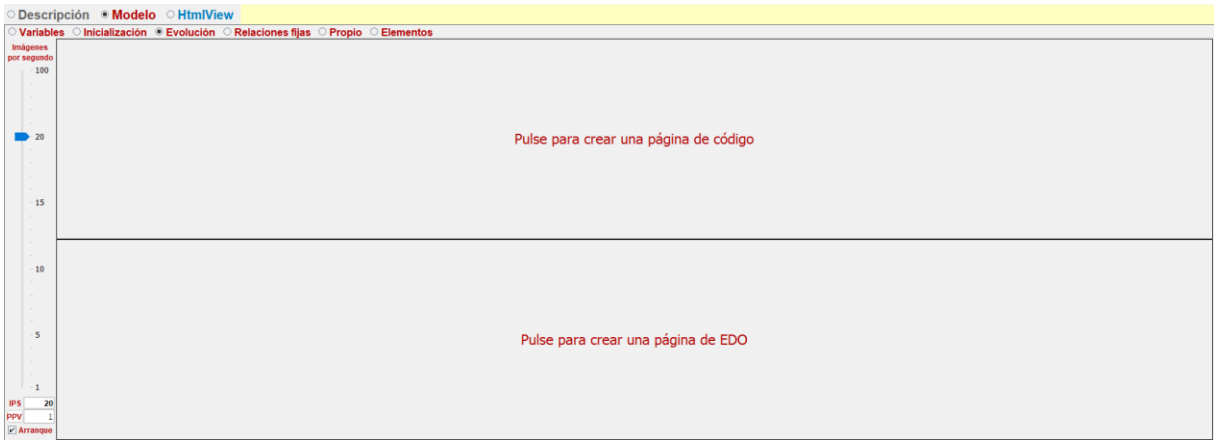

Figura 6-3. Pestaña de evolución en EJsS

6.2.1. Uso de ecuaciones diferenciales.

Para usar las ecuaciones diferenciales, se parte de la Ecuación 21 y de la dinámica que relaciona la posición de un motor con su tensión de entrada. Partiendo de la Ecuación 21 y aplicando la antitransformada de Laplace, se puede obtener la ecuación diferencial.

$$\alpha(t) = \frac{7}{5g} * \frac{d^2x(t)}{dt^2} + \frac{6\eta * \pi * R}{m * g} * \frac{dx(t)}{dt}$$

Ecuación 96

El siguiente paso, es despejar de la Ecuación 96 la derivada de mayor orden, obteniendo así la siguiente expresión.

$$\frac{d^2x(t)}{dt^2} = \frac{5g * \alpha(t)}{7} - \frac{30\eta * \pi * R}{7m} * \frac{dx(t)}{dt}$$

Ecuación 97

Para terminar con esta expresión, se debe tener en cuenta que la derivada de la posición será la velocidad de la bola, esto queda reflejado en la siguiente expresión:

$$\frac{dx(t)}{dt} = v(t)$$

Ecuación 98

Por lo que sustituyendo esta expresión en la Ecuación 97, se obtiene la siguiente expresión, que será usada en EJsS.

$$\frac{dv(t)}{dt} = \frac{5g * \alpha(t)}{7} - \frac{30\eta * \pi * R}{7m} * v(t)$$

Ecuación 99

El siguiente paso, es hacer el mismo proceso con la dinámica que relaciona la posición de un motor frente a su tensión de entrada, presente en la Ecuación 100.

$$G_{motor}(s) = \frac{\theta_1(s)}{M(s)} = \frac{k_m}{s * (\tau * s + 1)}$$

Ecuación 100

Aplicando la antitransformada de LaPlace, se obtiene la siguiente expresión.

$$m(t) = \frac{\tau}{k_m} * \frac{d^2\theta_1(t)}{dt^2} + \frac{1}{k_m} * \frac{d\theta_1(t)}{dt}$$

Ecuación 101

El siguiente paso, es simplificar, sabiendo que la derivada de la posición angular es la velocidad angular, tal y como viene reflejado en la Ecuación 29, además de despejar la derivada de mayor orden, obteniendo así la siguiente expresión.

$$\frac{d\omega(t)}{dt} = \frac{k_m * m(t) - \omega(t)}{\tau}$$

Ecuación 102

De esta manera, se estaría en disposición de introducir las ecuaciones diferenciales en EJsS, justo como queda reflejado en la Figura 2-1.

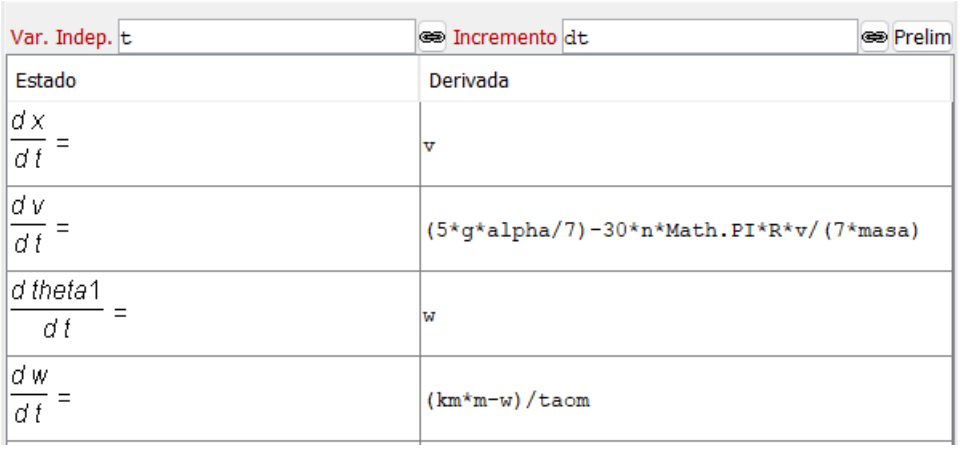

Figura 6-4. Declaración de ecuaciones diferenciales en EJsS

Es importante destacar, que en la figura anterior se puede observar que es en esta ventana donde se usan las variables temporales que se habían mencionado en el capítulo 6.1.

6.2.2. Uso de código.

El uso del código es exactamente el mismo que el desarrollado en el flujograma de la Figura 5-13, pero con una pequeña excepción, en este caso se deben marcar los límites físicos dentro del código, con el objetivo de que el funcionamiento del laboratorio virtual se asemeje lo máximo posible a la realidad. Para este laboratorio virtual, los límites físicos que se deben tener en cuenta son los siguientes:

- Salida del controlador esclavo: tal y como se ha descrito en los capítulos previos, el valor absoluto de esta variable puede encontrarse entre 0 y 100.
- Grado de inclinación del brazo del motor: para este laboratorio, el grado de giro del eje del motor, puede estar entre 90º y -90º, considerando 0 cuando el brazo del motor se encuentra paralelo en paralelo al suelo.
- Posición de la bola: como es lógico, el valor absoluto de la distancia de la bola al centro de la barra no puede ser mayor que la mitad de la longitud de esta.

6.3. Interfaz gráfica del laboratorio virtual.

El último paso para terminar el desarrollo del laboratorio virtual es crear la interfaz gráfica para permitir que el usuario pueda visualizar el funcionamiento del sistema barra-bola además de poder interactuar con él.

El diseño de dicha interfaz gráfica queda reflejado en la Figura 6-5.

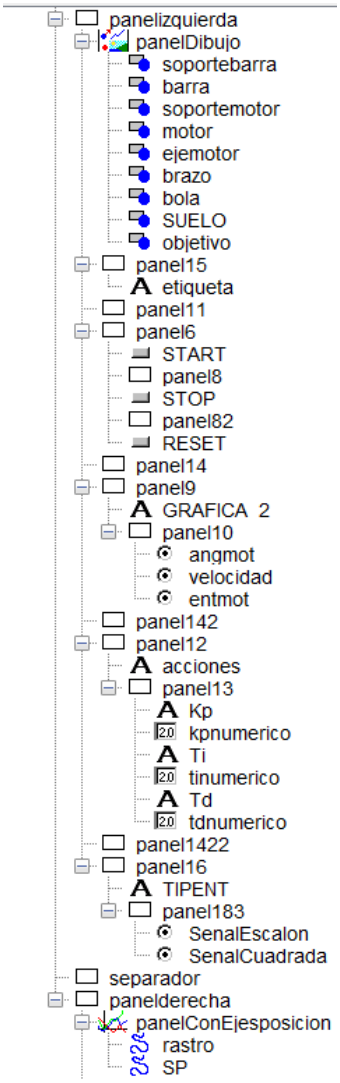

Figura 6-5. HTMLView para laboratorio virtual

Para que quede mejor detallado, en la Figura 6-6 y en la Figura 6-7 se puede observar cómo se han configurado los parámetros del elemento barra y brazo, usando las variables anteriormente descritas.

Figura 6-6. Configuración elemento barra

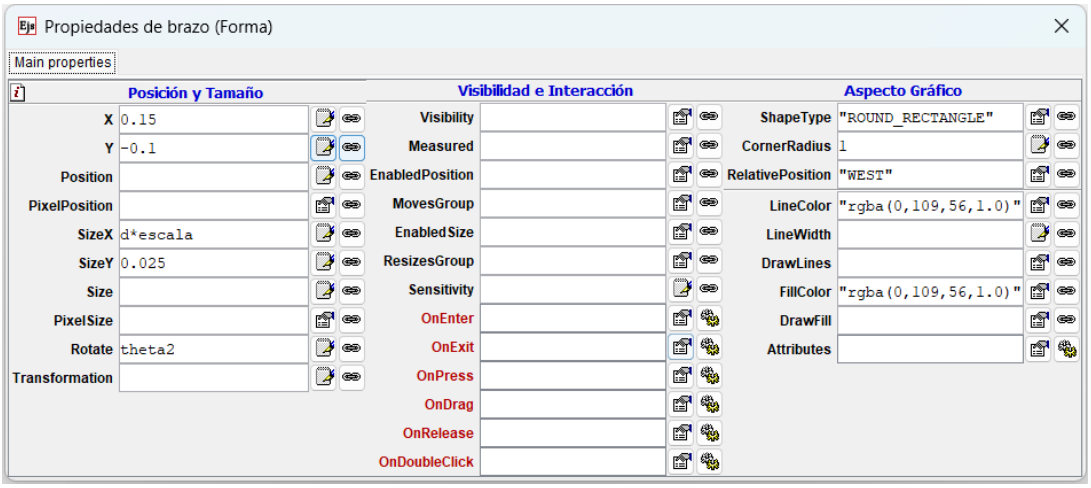

Figura 6-7. Configuración elemento brazo

Además, se puede observar el uso de la variable “escala”, la cual fue descrita en el capítulo 6.1 y que permite mantener la proporcionalidad en el tamaño de los elementos.

Por último, añadir que se han agregado una serie de funcionalidades a la interfaz gráfica, permitiendo así usar una gráfica que permite ver la posición de la bola y otra gráfica “variable”, en la cual el usuario puede elegir qué variable puede visualizar, además de la posibilidad de elegir un Set Point variable (señal cuadrada). El resultado del laboratorio virtual queda reflejado en la Figura 6-8.

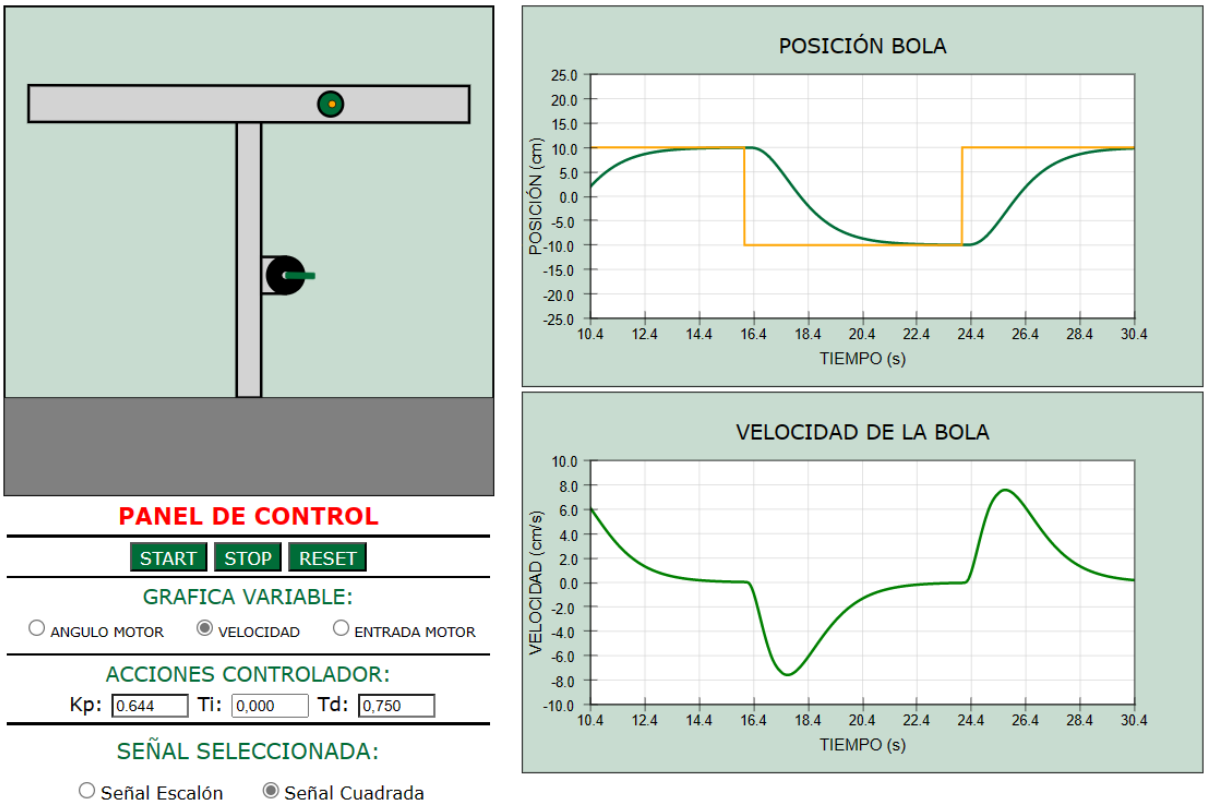

Figura 6-8. Resultado de la interfaz gráfica

Antes de finalizar este capítulo, es muy importante mencionar la importancia que tiene la gráfica “variable”, ya que permitirá al usuario visualizar si alguna variable ha saturado debido a los límites físicos del sistema, tal y como ya se mencionó en el capítulo de Límites físicos existentes en la planta real. En la Figura 6-9, puede visualizarse un ejemplo de saturación en el ángulo de giro del motor.

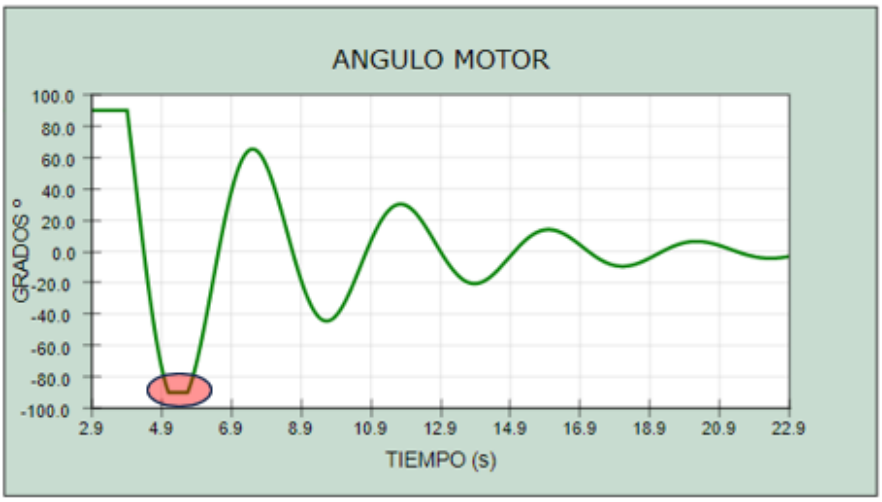

Figura 6-9. Saturación fruto de los límites físicos del sistema

7. Conclusiones y trabajos futuros.

Durante un periodo de 6 meses, se ha llevado a cabo el desarrollo de dos laboratorios online, uno remoto y otro virtual. Dicho desarrollo se inició con el modelado de los sistemas involucrados, el estudio de los diversos componentes que la forman y el cálculo teórico del lazo de control. Culminando de esta manera con el correcto funcionamiento del laboratorio remoto y la mayor semejanza posible entre el laboratorio virtual y la realidad.

A pesar de esto, durante el desarrollo de los laboratorios remoto y virtual han surgido una serie de problemas, los cuales han sido solventados para conseguir el correcto funcionamiento de ambos. Entre los principales problemas que surgieron se encuentran:

- Problemas de conexión entre servidor y cliente, para solventarlos se recibió ayuda del principal desarrollador del protocolo RIP, lo cual facilitó en gran medida el poder entender mejor el funcionamiento de dicho protocolo.
- Problemas con la configuración de la IP Cam, para solventarlos se tuvo que recurrir en gran medida al manual de usuario de dicho dispositivo además de la creación de un servidor NGINX, tal y como se ha descrito en el capítulo 5.2.2, para lo cual también se recibió una ayuda indispensable por parte del mismo desarrollador mencionado anteriormente.
- Problemas con el tiempo de ciclo del control del laboratorio remoto, para solventarlo el código del laboratorio remoto se optimizó en la medida de lo posible, reduciendo así el tiempo de ciclo del bucle, y obteniendo una respuesta mucho mejor.
- Robustez de la maqueta, ya que, en un principio, la cámara Raspberry Pi v2.1 se encontraba en una estructura separada de la estructura que soportaba al sistema barra-bola, esto hacía que las posiciones relativas entre estas dos estructuras pudiesen cambiar con mucha facilidad, por lo que se optó por unir todos los componentes en una misma estructura, siendo así la maqueta mucho más robusta.

Por otro lado, para aprovechar todo el potencial que ofrecen tanto el laboratorio remoto como el virtual, se han diseñado una serie de prácticas incrementales, permitiendo así al usuario, que en este caso es el alumno, partir desde lo más básico del sistema barra-bola para llegar hasta el sistema completo.

De esta manera se han desarrollado las siguientes prácticas:

- Práctica 0, Gamificación. Mediante el uso del laboratorio remoto, el alumno puede interactuar de manera manual con los distintos componentes del sistema para hacer el control en posición él mismo.
- Práctica 1, Modelado del motor. Mediante el uso de un laboratorio virtual, permite al alumno entender cómo se obtiene la dinámica del motor.
- Práctica 2, Control en posición del motor. Mediante el uso de un laboratorio virtual, permite al alumno llevar a cabo el control en posición del motor, pudiendo parametrizar a su elección el controlador. Además de tener una serie de preguntas que guían la práctica.
- Práctica 3, Modelado del sistema barra-bola. Mediante el uso de un laboratorio virtual, permite al alumno entender cómo se obtiene la dinámica del sistema barra-bola.
- Práctica 4, Control en posición de barra-bola. Mediante el uso de un laboratorio virtual, permite al alumno llevar a cabo el control en posición del sistema barra-bola, pudiendo parametrizar a su elección el controlador. Además de tener una serie de preguntas que guían la práctica.
- Práctica 5, Control en posición de sistema Barra-Bola Real. Mediante el uso de un laboratorio remoto, permite al alumno llevar a cabo el control en posición del sistema barra-bola, pudiendo parametrizar a su elección el controlador.

Dichas prácticas se pueden visualizar en el ANEXO 1. Además, han sido utilizadas en la asignatura de Ingeniería de Control impartida en 3º de Ingeniería Electrónica Industrial durante el curso 2023-2024, siendo bien recibidas por los alumnos como una herramienta más para complementar su formación.

Respecto a posibles trabajos futuros que pueden realizarse para mejorar el funcionamiento de los laboratorios, se encuentran los siguientes:

- Creación de una PCB para la integración de todos los dispositivos hardware, mejorando de esta manera la robustez de las conexiones entre dispositivos.
- Añadir más funciones SCORM, ya que, con la implementación actual, en la plataforma online de educación queda registrada la puntuación obtenida por el usuario, sin embargo, no queda registrada la respuesta que ha dado, por lo que sería interesante obtener este tipo de información.

REFERENCIAS

[1] Clough, M.P. (2002). Using the laboratory to enhace student learning. *Learning Science and the Science of Learning, JANUARY 2002*, 85-94.

[2] Corter, J.E., Nickerson, J. V., Esche, S. K., Chassapis, C., IM, S., & Ma, J. (2007). Constructing reality: A study of Remote, Hands-On, and Simulated Laboratories. *ACM Transactions on Computer-Human Interaction, 14(2), 7-es*. https://doi.org/10.1145/1275511.1275513

[3] B. Balamuralithara and P. C. Woods, "Virtual laboratories in engineering education: the simulation lab and remote lab", *Computer Applications in Engineering Education*, vol.17, no. 1, 2009, doi: 10.1002/cae.20186

[4] Luis de la Torre, Jesús Chacón, and Dictino Chaos, "Specification of the Remote Interoperaibility Protocol for Online Laboratories". Accessed: Jan. 20, 2024. Url: https://github.com/UNEDLabs/rip-spec/blob/master/RIP_Specification.pdf

[5] F. Esquembre, "Easy Java Simulations: A software tool to create scientific simulations in Java." *Computer Physics Communications*, vol. 156, no. 2, 2004, doi: 10.1016/S0010-4655(03)00440-5.

[6] "Leyes de Newton – Wikipedia, la enciclopedia libre". Url: https://es.wikipedia.org/wiki/Leyes_de_Newton.

[7] "Rodadura – Wikipedia, la enciclopedia libre". Url: https://es.wikipedia.org/wiki/Rodadura

[8] "Ley de Stokes – Wikipedia, la enciclopedia libre". Url: https://es.wikipedia.org/wiki/Ley_de_Stokes

[9] Castaño Giraldo, Sergio Andrés. Control Automático Educación. *Modelado de Motor DC.* Accessed: Jan. 20, 2024. Url: https://controlautomaticoeducacion.com/analisis-de-sistemas/modelo-de-motor-dc/#Modelo_Electromecanico

[10] "Leyes de Kirchhoff – Wikipedia, la enciclopedia libre". Url: https://es.wikipedia.org/wiki/Leyes_de_Kirchhoff

[11] “Diagramas de bloques de Simulink - MathWorks”. Accessed: Jan. 20, 2024. Url: https://es.mathworks.com/help/simulink/gs/simulink-block-diagrams.html

[12] “Transformada de Laplace – Wikipedia, la enciclopedia libre”. Url: https://es.wikipedia.org/wiki/Transformada_de_Laplace

[13] Castaño Giraldo, Sergio Andrés. Control Automático Educación. *Sistemas Dinámicos de Primer Orden.* Accessed: Jan. 20, 2024. Url: https://controlautomaticoeducacion.com/control-realimentado/sistemas-dinamicos-de-primer-orden/.

[14] Pololu. 70:1 Metal Gearmotor 37Dx70L mm. Accessed: Jan. 20, 2024. Url: https://www.pololu.com/product/2825

[15] “Raspberry Pi – Wikipedia, la enciclopedia libre”. Url: https://es.wikipedia.org/wiki/Raspberry_Pi

[16] DFROBOT. “L298 Dual H-Bridge Motor Driver”. Accessed: Jan 20, 2024. Url: https://wiki.dfrobot.com/MD1.3_2A_Dual_Motor_Controller_SKU_DRI0002

[17] Acciones Básicas de Sistemas de Control. Accessed: Jan 20, 2024. Url: https://dademuchconnection.wordpress.com/2018/04/05/acciones-basicas-de-sistemas-de-control/

[18] Castaño Giraldo, Sergio Andrés. Control Automático Educación. *Control en cascada.* Accessed: Jan. 20, 2024. Url: https://controlautomaticoeducacion.com/control-realimentado/control-en-cascada/.

[19] “Cómo crear un certificado SSL autofirmado para Apache en Ubuntu 18.04 | DigitalOcean”. Accessed: Jan 20, 2024. Url: https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-18-04-es

[20] “Distancia euclidiana – Wikipedia, la enciclopedia libre”. Url: https://es.wikipedia.org/wiki/Distancia_euclidiana

[21] PROGRAMO ERGO SUM. *"Arranque automático en Raspbian"*. Accessed: Jan 20, 2024. Url: https://www.programoergosum.es/tutoriales/arranque-automatico-en-raspbian/.

[22] AXIS M3044-V NETWORK CAMERA – *Manual de usuario*. Accessed: Jan. 21, 2024. Url: https://help.axis.com/es-es/axis-m3044-v

[23] Kinsta - *¿Qué es Nginx y Cómo Funciona?* Accessed: Jan. 21, 2024. Url: https://kinsta.com/es/base-de-conocimiento/que-es-nginx/

[24] "SCORM: qué es y por qué es clave en e-learning." https://www.homuork.com/es/scorm-que-es-y-por-que-es-clave-en-el-e-learning_226_102.html

[25] "Easy Java Simulations Wiki | Main / EJSUserInterface." Accessed: Jan. 21, 2024. Url: https://www.um.es/fem/EjsWiki/Main/EJSUserInterfarce

ANEXO 1: Guiones de prácticas diseñadas y soluciones.

En este primer anexo, se presentan los distintos guiones de las prácticas que se han desarrollado para permitir al alumno la comprensión de manera incremental del sistema barra-bola además de sus respectivas soluciones.

1. Práctica 0: CONTROL MANUAL DEL SISTEMA BARRA-BOLA.

1.1. Introducción.

Esta práctica consiste en llevar a cabo el control en posición de un sistema barra-bola, ubicado en un laboratorio remoto, de manera manual, facilitando así al alumno la posibilidad de conocer el sistema con el que se trabajará en las siguientes prácticas.

1.2. Objetivos.

El principal objetivo de esta práctica es que el alumno visualice los distintos elementos que constituyen el sistema barra-bola, y que pueda interactuar con este de manera manual intentando llevar la bola a la posición central de la barra.

Para ello, la práctica se presenta al estudiante en forma de gamificación, haciendo así que el alumno entienda cómo funciona el sistema, y además comprenda que llevar a cabo un control manual presenta desventajas frente al control automático.

1.3. Manual de usuario.

Al acceder a la práctica el alumnado tendrá la interfaz que se ilustra en la Figura A1-1. Se observa que en: (a) se presenta un vídeo en tiempo real del sistema barra-bola; (b) una gráfica (aún vacía) de la posición de la bola en función del tiempo; (c) se presenta el panel de control, donde de momento únicamente aparece un botón para activar la iluminación; (d) un panel del estado del juego que se encuentra actualmente vacío.

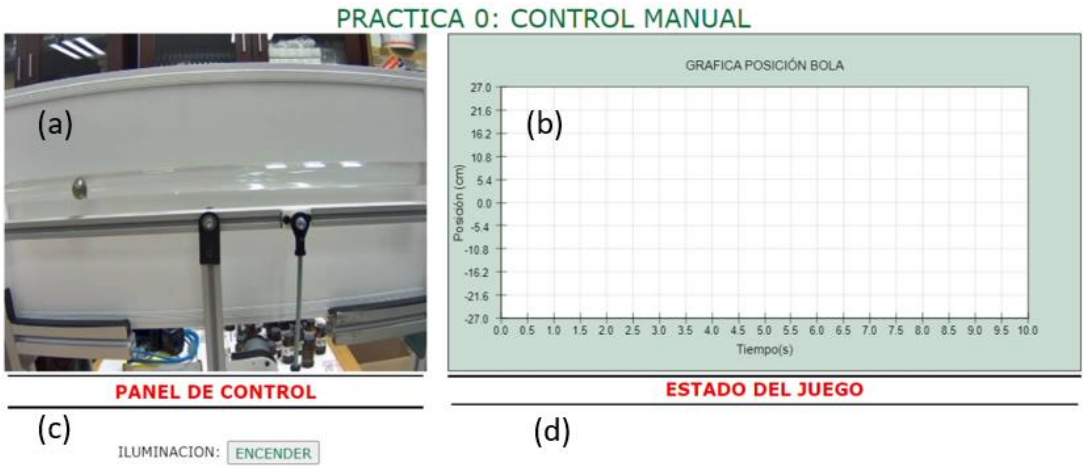

Figura A1-1. Interfaz Gráfica de la práctica 0: (a) video streaming; (b) gráfica de posición en tiempo real de la bola; (c) zona de sistema de iluminación; (d) Estado del juego -todavía sin comenzar-

Los pasos para poder realizar la práctica son los siguientes:

1. <u>Encender panel.</u> Lo primero que se debe hacer, es encender el panel de iluminación. Esto es necesario para poder detectar la posición de la bola mediante una cámara que se encuentra en el laboratorio. Como se puede observar en la Figura A1-2, tras encender el panel de iluminación, se hará visible el botón (PM), el cual permite llevar a cabo la puesta en marcha.

Figura A1-2. Encendido del sistema de iluminación. El sistema está listo para la puesta en marcha

2. <u>Puesta en marcha.</u> Este paso tiene como finalidad definir siempre un mismo punto de partida de la bola, en este caso, el extremo izquierdo. Este estado se activa una vez que se pulsa el botón “PM”. Cuando la bola alcanza su posición de inicio el sistema lo avisará haciendo que el panel

de iluminación parpadee. Además, como se observa en la Figura A1-3, se harán visibles varios campos:

- **Velocidad:** dos botones que permiten subir (+VEL) y bajar (-VEL) la velocidad de giro del motor, además de un campo numérico que indica la velocidad en %.
- **START GAME:** botón que tras ser pulsado hará que comience el juego.
- **Tiempo restante:** indica el tiempo restante para completar el juego, inicialmente su valor es de 60 segundos y comenzará a descender una vez pulsado “START GAME”, si su valor llega a 0 y la bola no ha llegado a su objetivo no se habrá conseguido el objetivo.
- **Estado del juego:** indica el estado actual del juego, existiendo 4 posibilidades: “JUEGO EN ESPERA”, “JUEGO EN MARCHA”, “HAS GANADO” y “HAS PERDIDO”.

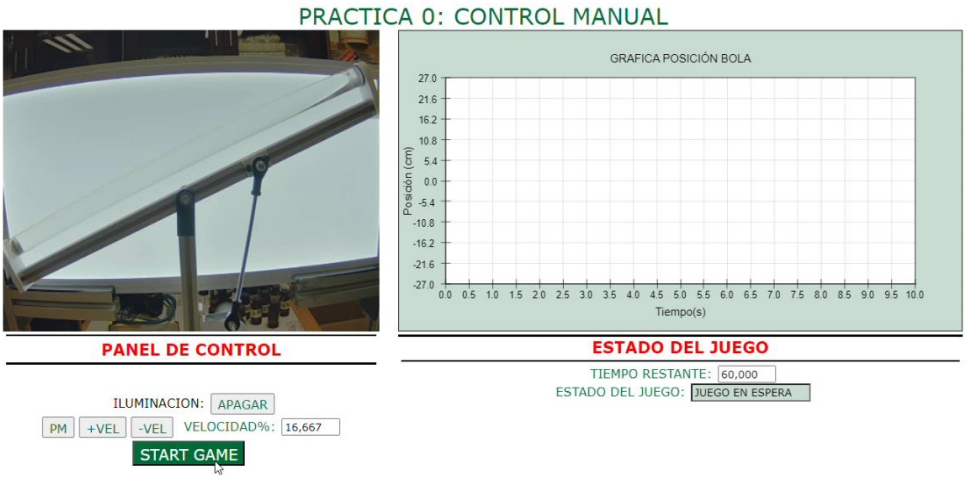

Figura A1-3. Interfaz de la práctica 0 cuando el sistema está listo para comenzar el juego

3. Inicio del juego: tras pulsar “START GAME”, como se ha explicado previamente, el temporizador de 60 segundos se activará, por otro lado, en la parte superior del panel de control, se harán visibles dos botones “IZQ” y “DER”, que permitirán mover la barra hacia la izquierda o la derecha respectivamente.

Figura A1-4. Interfaz de la práctica 0 cuando el/la alumno/a está jugando

4. Fin del juego: si se consigue el objetivo, se obtendrá un mensaje que informará del éxito de la actividad además del tiempo que sea requerido para conseguirlo, tal y como se ha reflejado en la Figura A1-5. En caso de no conseguir dejar la bola en la posición central de la barra, también se recibirá un mensaje. En cualquiera de los dos casos, se presentan dos opciones, la primera es volver a pulsar “PM”, para volver a realizar la actividad hasta estar conforme con el resultado, la segunda es pulsar “OBTENER LA PUNTUACIÓN”, para que se asigne una puntuación al tiempo requerido para completar la actividad, pudiendo así abandonar la actividad.

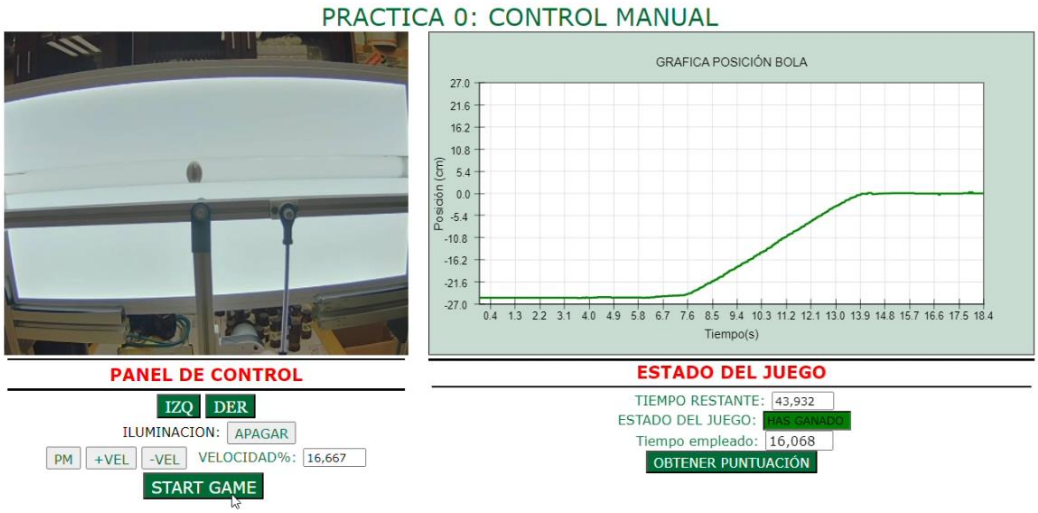

Figura A1-5. Interfaz de la práctica 0 cuando el/la alumno/a ha alcanzado el objetivo de la práctica- Mover la bola al centro de la barra-. Nótese que el centro de la barra se considera la posición 0, de ahí que el rango de movimiento de la bola sea de -27 cm a 27 cm (teniendo la barra una longitud de 54 cm)

1.4. Conclusiones.

El estudiante debe de terminar la práctica habiendo entendido cómo funciona el sistema barra-bola y que llevar a cabo un control manual de un sistema no proporciona los mejores resultados, ya que estos se verán influidos por factores como la habilidad del usuario, por lo que no es una manera eficiente ni exacta de llevar a cabo el control.

2. Práctica 1: MODELADO EXPERIMENTAL DINÁMICA MOTOR.

2.1. Introducción.

El modelado experimental es un pilar fundamental de la Ingeniería de Control, ya que este se trata de una gran herramienta que permite obtener una función matemática que relaciona las variables de salida del sistema con las variables de entrada del mismo.

Esta práctica consiste en obtener la dinámica de la posición del motor teniendo como dato de partida la respuesta en velocidad del mismo, facilitando al alumno la posibilidad de visualizar su respuesta calculada frente a la respuesta correcta.

2.2. Objetivos.

El objetivo fundamental de esta práctica es que el alumno se inicie en el modelado experimental de dinámicas. Para ello el estudiante deberá de llevar a cabo los cálculos necesarios para obtener la dinámica y posteriormente visualizar su respuesta calculada. Por otro lado, esta práctica servirá al alumno para próximamente, entender mejor cómo funciona el lazo de control del motor.

2.3. Funcionamiento.

Para comenzar, se debe de conocer el equipo que se está usando, en este caso, se trata del motor **70:1 Metal Gearmotor 37Dx70L**, el cual, para ser excitado, puede recibir una tensión entre 0 y 12 voltios.

Cómo es excitado, no afectará al modelado experimental, y es un tema que será tratado en la siguiente práctica.

Ahora que se conoce el motor, y se sabe cuál es el objetivo, se está en disposición de obtener la dinámica de la velocidad del motor, pero primero se debe de entender cómo funcionan los bloques, ya que este motor consta de un encoder, el cual, permite contar pulsos. Tal y como se puede observar en la Figura A1-6.

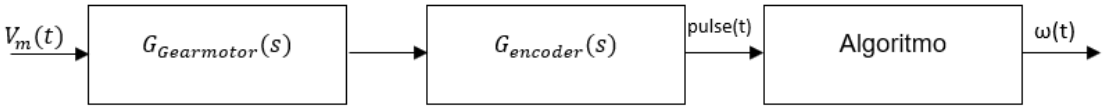

Figura A1-6. Conjunto de bloques Gearmotor, Encoder y Algoritmo

La señal $V_m(t)$ será la tensión que varía entre 0-12 voltios, si se excita al motor con esta señal, este girará, y gracias al encoder, se podrán contar los pulsos, obteniendo así la señal pulse(t), lo único que quedaría, sería obtener la velocidad del motor, esto se hace mediante un algoritmo que calcula incrementos de pulsos y divide estos incrementos de posición entre incrementos temporales.

Para finalizar se debe de pasar la velocidad de pulsos por segundo (pps) a revoluciones por segundo (rps), para realizarlo, de la ficha técnica se obtiene que una revolución tiene 4480 pulsos, y de esta manera se obtiene la señal ω(t) que es la velocidad del motor en rps.

Los tres bloques mencionados arriba, quedarán simplificados en uno solo de la siguiente manera:

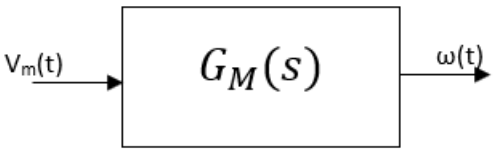

Figura A1-7. Conjunto de bloques del motor simplificados

Y la gráfica que se obtiene tras excitar el motor con incremento de 0 a 12 V en continua queda reflejada en la Figura A1-8.

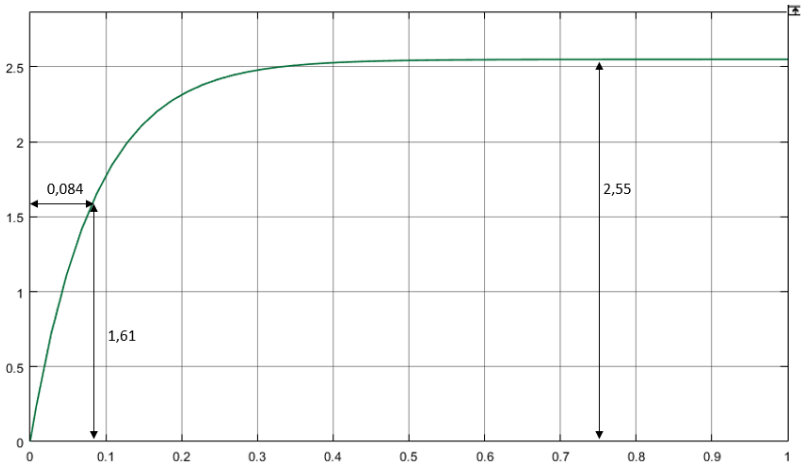

Figura A1-8. Velocidad del motor en rps respecto al tiempo tras variar la tensión entre 0-12 V

Una vez obtenida esta dinámica, para obtener la dinámica de la posición, que es la que de verdad interesa, sólo se debe de tener en cuenta que la velocidad es la derivada de la posición.

2.4. Uso de la interfaz.

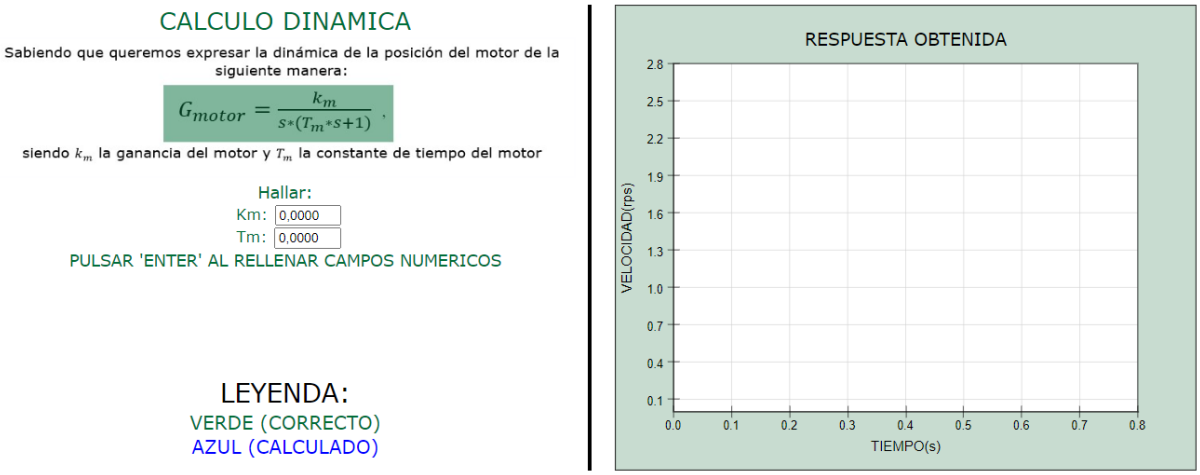

Figura A1-9. Interfaz gráfica de la práctica 1

Para usar la interfaz, tan solo se deben introducir los valores calculados según el formato indicado, tras esto aparecerá el botón “RESULTADOS” y después de ser pulsado se podrá visualizar la respuesta correcta y la forma de la respuesta calculada.

2.5. Conclusiones.

Tras realizar la práctica, el estudiante debe de ser capaz de entender cómo se excita un motor de corriente de continua, obteniendo así su respuesta y a partir de esta, conseguir la dinámica del motor.

Además, el alumno debe de comprender qué efecto tiene la constante de tiempo y la ganancia del motor sobre la señal de salida del sistema, es decir, qué ocurre si estos parámetros aumentan o disminuyen su valor.

2.6. Solución.

Para llegar al resultado se siguen los pasos detallados en el capítulo Modelado matemático del motor mediante su respuesta. Llegando así a la solución de:

$$k_m = 0.2125; \qquad T_m = 0.084;$$

3. Práctica 2: CONTROL EN POSICIÓN DEL MOTOR DC.

3.1. Introducción.

Esta práctica consiste en obtener los parámetros de un controlador PID, con la finalidad controlar la posición de un motor DC. Para ello, el alumno tendrá como herramienta una réplica virtual del motor a controlar. Este lazo de control será usado posteriormente para controlar la posición de la bola dentro de un sistema barra-bola.

3.2. Objetivos.

El objetivo de esta práctica es que el alumno aprenda y comprenda cómo obtener los parámetros de un controlador, y cómo estos afectan a la respuesta del sistema. Para facilitar la comprensión del estudiante, se usa como herramienta una réplica virtual del motor, pudiendo así visualizar la respuesta del motor con los parámetros que ha calculado. Para guiar la práctica, dentro de la interfaz, el alumno tendrá una serie de cuestiones a responder.

Por otro lado, esta práctica permite al alumno entender mejor cómo funciona el motor y cómo está formado el lazo de control.

3.3. Funcionamiento.

El lazo de control completo es que se muestra en la figura:

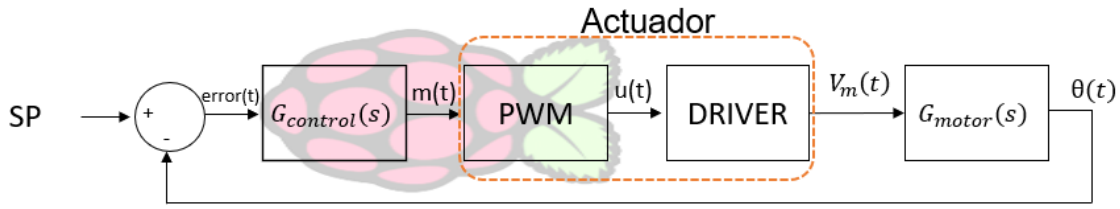

Figura A1-10. Lazo de control en posición de un motor DC a través de un microprocesador Raspberry Pi

Según ilustra la Figura A1-10, el algoritmo de control se ejecuta en un microprocesador, concretamente en una Raspberry Pi, dispositivo monoplaca con un sistema operativo Linux.

La ecuación diferencial que define el comportamiento del algoritmo de control es la siguiente:

$$m(t) = K_p * \left[1 + T_d \frac{derror(t)}{dt} + T_i \int error(t)dt\right]$$

La dinámica del controlador tiene como entrada el error y como salida la señal m(t).

El valor de m(t) indica el % de tiempo que una señal periódica se encuentra en su valor máximo (Duty Cycle), por lo que dicho valor podrá oscilar entre 0 y 100%. La Figura A1-11, muestra varios ejemplos. Véase que, si se tiene un Duty Cycle de 50%, la señal se encontrará la mitad del periodo a nivel alto y la otra mitad a nivel bajo.

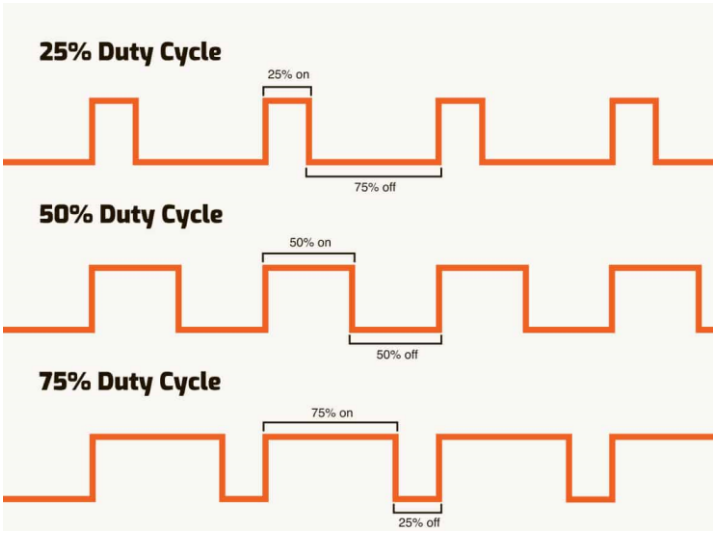

Figura A1-11. Ejemplo de valores de la variable Duty Cycle

Como ya se sabe, el motor debe de ser excitado con tensión continua (0-12V) por lo que la señal m(t) debe de ser transmitida y transformada. Esta labor es la que desempeña normalmente un actuador dentro del lazo de control. En este caso, el actuador está formado por dos módulos: un bloque PWM (Pulse Width Modulation) y un Driver.

Para que una señal pueda ser transmitida tiene que ser modulada y esto es precisamente lo que realiza el bloque PWM. Dicho bloque recibe el Duty Cycle y obtiene el promedio de la tensión de salida a lo largo del tiempo. El valor de dicha tensión puede ser como máximo 3.3V ya que es lo que permite generar la fuente de alimentación integrada de la Raspberry. Dicha tensión no es compatible con la tensión necesaria para excitar el motor DC, de ahí la necesidad de incorporar un Driver, en este caso se trata del Dual H Bridge L298N. Dicho driver escala la tensión que recibe a la tensión necesaria para excitar el motor ($V_m(t)$) siendo esta la variable de proceso manipulada. La Figura A1-12 muestra un ejemplo.

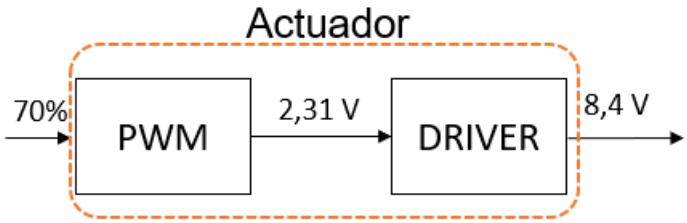

Figura A1-12. Ejemplo de la transmisión y escalado de la tensión de entrada del motor DC

Si la señal m(t) es del 70%, esto producirá que la salida del bloque PWM sea de 2.31 V, y esta señal será escalada a través del DRIVER, obteniendo un $V_m(t)$ de 8.4 V de tensión continua que excitarán el motor. Como se ha comentado, el conjunto de ambos bloques desempeña el papel de actuador cuya dinámica se puede aproximar a una ganancia:

$$G_{PWM}(s) = \frac{V_{pi}(s)}{M(s)} = \frac{3.3 - 0}{100 - 0} = \frac{3.3}{100}\left(\frac{v}{\%}\right);$$

$$G_{Driver}(s) = \frac{V_m(s)}{V_{pi}(s)} = \frac{12 - 0}{3.3 - 0} = \frac{12}{3.3}\left(\frac{v}{v}\right);$$

$$G_a(s) = \frac{V_m(s)}{M(s)} = G_{PWM}(s) * G_{Driver}(s) = \frac{12}{100} = 0.12\left(\frac{v}{\%}\right)$$

Finalmente, la dinámica del motor que define la evolución de la posición en función de la tensión aplicada, hallada en la práctica, es:

$$G_{motor}(s) = \frac{\theta(s)}{V_m(s)}\frac{k_m}{s * (\tau_m * s + 1)}$$

Siendo k_m=0.2125 y τ_m=0.084.

Como se comentó en la práctica anterior, dicho bloque, incluye el motor en sí, un encoder y un algoritmo que convierte la salida de dicho encoder a revoluciones.

3.4. Uso de la interfaz.

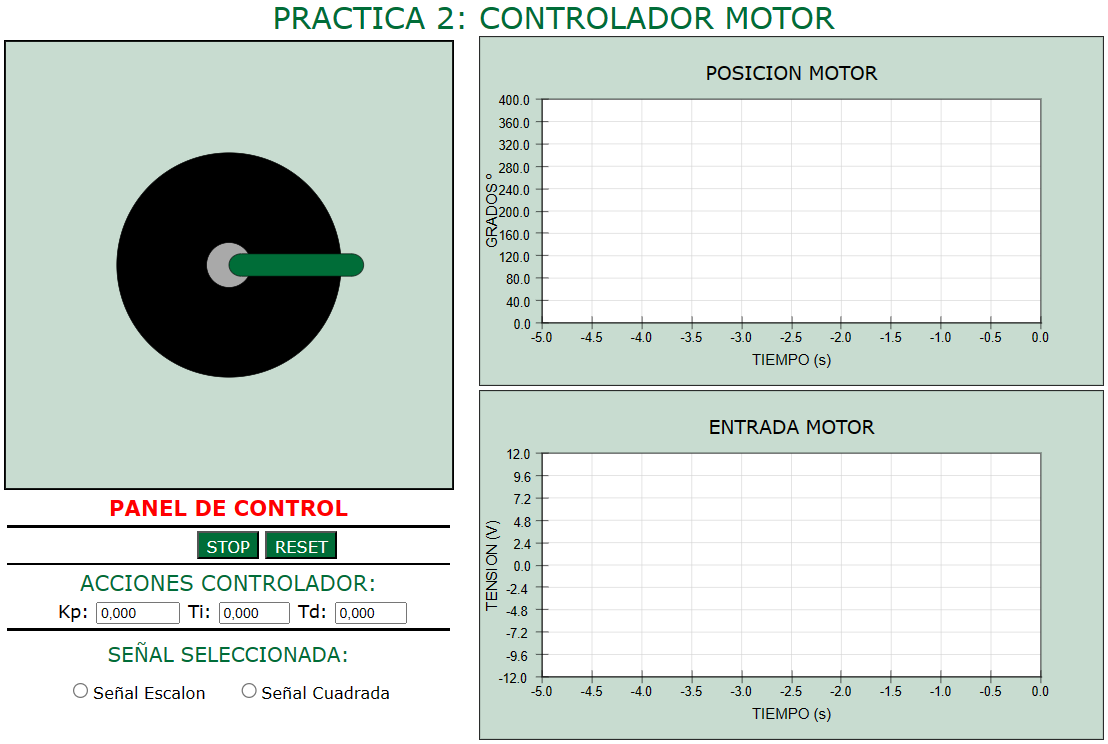

Figura A1-13. Interfaz gráfica de la práctica 2

Lo primero que se debe de tener claro, es la expresión de la dinámica del controlador. Al tratarse de un PID, la dinámica es la siguiente:

$$G_{control}(s) = k_p * (1 + T_d * s + \frac{T_i}{s})$$

Una vez calculados los parámetros del controlador, ya sean los solicitados en las preguntas o los que el estudiante desee probar, se pueden introducir en el apartado de “ACCIONES CONTROLADOR” del panel de control de la interfaz.

Tan pronto como estos se introduzcan, al lado del botón “STOP” aparecerá otro botón “START”, si se pulsa, la simulación comenzará. En cualquier momento, esta simulación se puede parar pulsando “STOP” y reanudarse de nuevo pulsando “START”, además se puede resetear la simulación pulsando “RESET”.

Dentro del panel de control, existe otra opción, que permite cambiar la señal de consigna (set point), teniendo dos opciones, una señal escalón y una señal cuadrada, esta opción se podrá cambiar en plena simulación. Si no se selecciona ninguna, de manera predeterminada, se usará una entrada señal escalón.

Es importante, una vez respondidas todas las preguntas, pulsar el botón de “ENVIAR”, para que la nota del estudiante quede grabada.

3.5. Conclusiones.

Tras realizar la práctica, el alumno debe de ser capaz de razonar qué tipo de controlador necesita según las especificaciones que se exijan, además de haberse familiarizado a trabajar con el lugar de las raíces.

El alumno, debe de comprender cómo afectan los parámetros del controlador a la respuesta, y puede llevar a cabo pruebas para ver cómo es la respuesta cuando el sistema es subamortiguado, críticamente amortiguado y sobreamortiguado.

3.6. Solución.

En la Figura A1-14, se pueden observar las respuestas correctas a las cuestiones que debe responder el alumno.

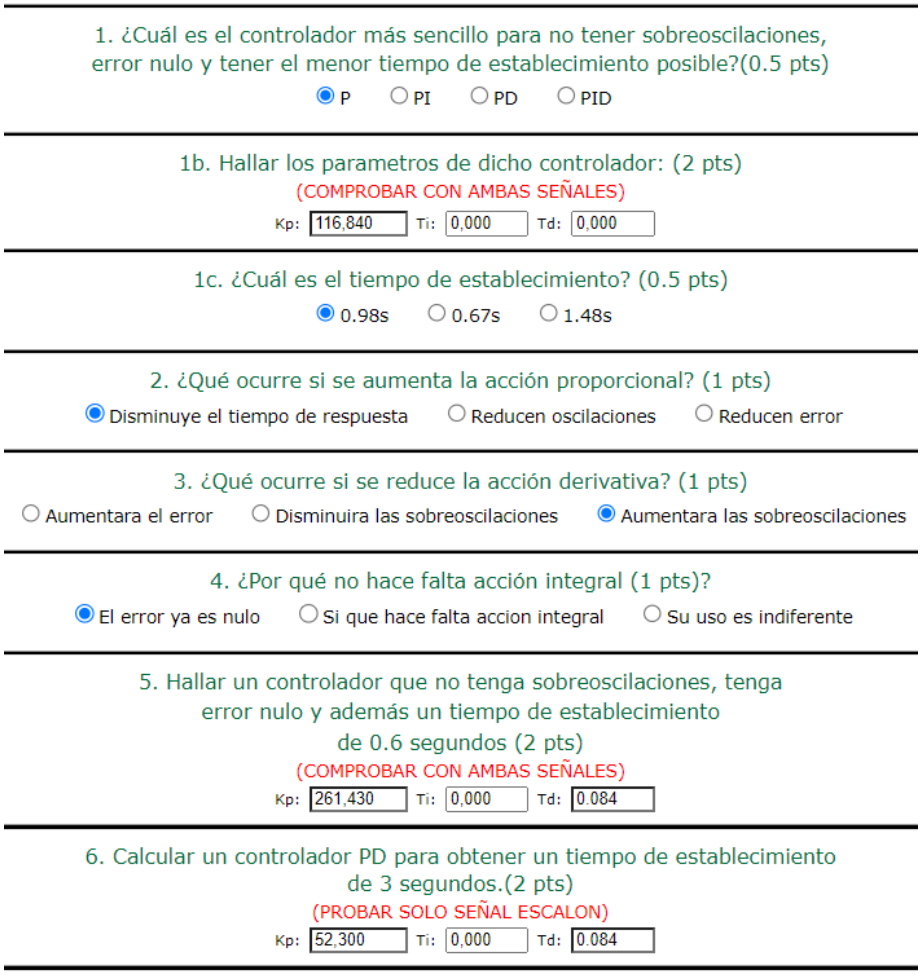

Figura A1-14. Soluciones de la práctica 2

4. Práctica 3: MODELADO MATEMÁTICO DINÁMICA BARRA-BOLA.

4.1. Introducción.

El modelado matemático es un pilar fundamental de la Ingeniería de Control. El modelado matemático consiste en la aplicación de leyes físicas con objeto de describir el comportamiento de un sistema. Por tanto, en él se manejan ecuaciones diferenciales que pasadas a un dominio algebraico permiten definir funciones de transferencia o también conocidas como dinámicas de sistemas.

En concreto, esta práctica tiene como finalidad la obtención de la dinámica de un sistema barra-bola. Dicha función de transferencia representará cómo evoluciona la posición de la bola en función del ángulo de inclinación de la barra. Para ello, será fundamental aplicar la 2ª Ley de Newton, teniendo como dato de partida las distintas fuerzas que se ven involucradas.

Finalmente, el alumno podrá corroborar que ha calculado de forma correcta la dinámica. Se le facilita la posibilidad de visualizar cómo evoluciona la posición de la bola según la dinámica calculada. En la misma gráfica también se visualiza la evolución de la posición de la bola correcta.

4.2. Objetivos.

El objetivo fundamental de esta práctica es que el alumno se inicie en el modelado matemático de dinámicas. Para ello, el estudiante deberá de llevar a cabo los cálculos necesarios para obtener la dinámica y posteriormente visualizar la respuesta calculada.

4.3. Cálculo de la Dinámica del sistema barra-bola.

La dinámica que se busca obtener en esta práctica tiene como entrada el ángulo de inclinación de la barra α(t), mientras que la salida se trata de la posición de la bola x(t).

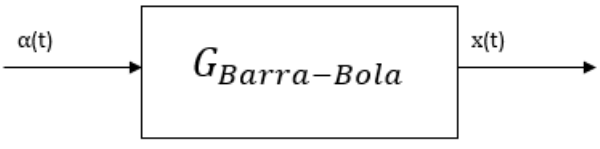

Figura A1-15. Bloque de la dinámica barra-bola

Hay que tener en cuenta que se trata de un sistema con una dinámica muy rápida, por lo que, con objeto de reducir la velocidad de desplazamiento de la bola, se ha introducido glicerina rebajada con agua en el interior de la barra.

A continuación, se presentan las distintas fuerzas implicadas:

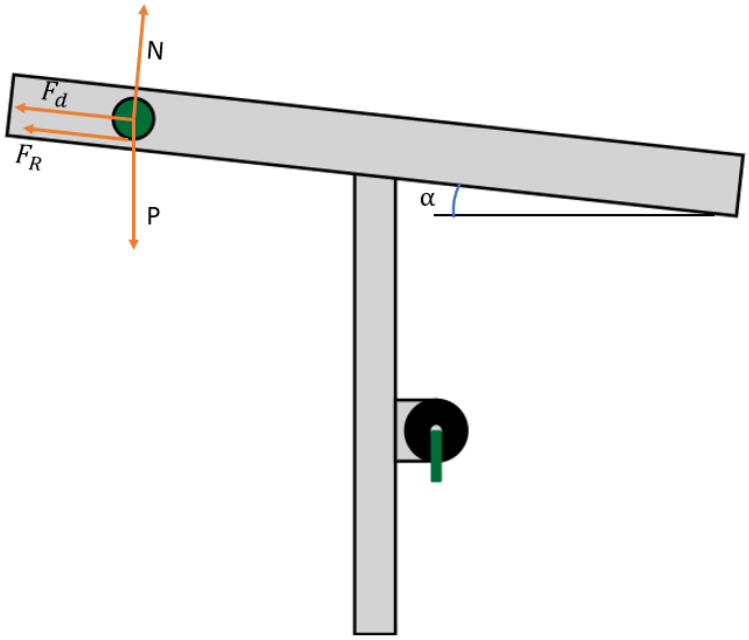

Figura A1-16. Esquema y aplicación 2ª Ley de Newton al sistema barra-bola

- **F_d(t):** fuerza de arrastre del fluido, que es la fricción entre un objeto sólido y el fluido por el que se mueve.
- **F_R(t):** fuerza de rozamiento.
- **P(t):** peso de la bola, este se descompone en P_x(t) y P_y(t).
- **N(t):** fuerza normal.

Para definir el comportamiento del sistema, es necesario aplicar la 2ª Ley de Newton: $\sum \vec{F}$=m*a(t) tanto en el eje X como en el Eje Y.

Comenzando por el eje Y, que será igual a cero, ya que inicialmente la bola se encuentra en reposo y por tanto no hay aceleración, $\sum \vec{F}_y$=0.

A continuación, se debe de llevar a cabo, el sumatorio de fuerzas en el eje X, $\sum \vec{F}_x$=m*a(t).

Para llevar a cabo el desarrollo matemático se facilitan las siguientes igualdades:

$$a(t) = R * a_{\alpha}(t)$$

$$F_R(t) * R = J_{bola} * a_{\alpha}(t)$$

$$J_{bola} = \frac{2}{5} * m * R^2$$

$$F_d(t) = 6 * \eta * \pi * R * v_x(t)$$

Siendo:

- a(t): aceleración lineal.
- a_{α}(t): aceleración angular.
- R: radio de la bola.
- J_{bola}: momento de inercia de una esfera sólida.
- m: masa de la bola.
- η: viscosidad de la mezcla de glicerina y agua.
- v_x(t): velocidad lineal de la bola.

Considerando:

- g=9.81 m/s^2.
- R=0.025 m.
- m=0.05 kg.
- η=0.418 (kg/(m*s))

Tras llevar a cabo el desarrollo matemático y sustituir el valor de las constantes, se obtendrá una dinámica con la siguiente estructura:

$$G_{Barra-Bola}(s) = \frac{a}{s * (s + p_1)}$$

Consiguiendo así la dinámica del sistema barra-bola.

4.4. Uso de la interfaz.

Figura A1-17. Interfaz gráfica de la práctica 3

Una vez calculada la dinámica, y sabiendo que la dinámica objetivo tendrá la siguiente estructura:

$$\frac{X(s)}{\alpha(s)} = \frac{a}{s*(s+p_1)}$$

Tan solo se debe de introducir los valores en sus respectivos campos numéricos, tras haberlos introducido, se hará visible el botón de “RESULTADOS”, y para finalizar la práctica tan solo habrá que pulsar dicho botón, de esta manera se quedará grabada la nota del estudiante.

Tras pulsar el botón de “RESULTADOS”, en la gráfica de la derecha se puede visualizar la respuesta en velocidad del sistema barra-bola producida por una entrada escalón unitaria calculada por el estudiante, frente a la respuesta correcta producida por la misma entrada escalón.

Por lo que la dinámica de la respuesta calculada que se visualizará se trata de:

$$s * G_{Barra-Bola} = \frac{a}{s + p_1}$$

4.5. Conclusiones.

Tras realizar la práctica, el estudiante debe de haber entendido cómo se obtiene la dinámica del sistema barra-bola mediante el modelado matemático.

4.6. Solución.

Para llegar al resultado se siguen los pasos detallados en el capítulo Modelado del sistema barra-bola. Llegando así a los siguientes resultados:

$$a = 7; \qquad p = 2.814;$$

5. Práctica 4: CONTROL DEL SISTEMA BARRA-BOLA. LAB VIRTUAL.

5.1. Introducción.

Esta práctica consiste en obtener los parámetros de un controlador PID, con la finalidad de llevar a cabo el control del sistema barra-bola. Para el lazo de control que se va a usar, se requiere del lazo de control que se desarrolla en la práctica 2. Además, el estudiante tiene como herramienta una réplica virtual del sistema barra-bola.

5.2. Objetivos.

El objetivo de esta práctica es que el alumno aprenda y comprenda cómo obtener los parámetros de un controlador, y cómo estos afectan a la respuesta del sistema. Para facilitar la comprensión del estudiante, se usa como herramienta una réplica digital del sistema barra-bola, pudiendo así visualizar la respuesta del sistema con los parámetros que ha calculado. Para guiar la práctica, dentro de la interfaz, el alumno tendrá una serie de cuestiones a responder.

Por otro lado, esta práctica permite al alumno entender mejor cómo funciona el sistema y cómo está formado el lazo de control.

5.3. Funcionamiento.

El lazo de control completo es el siguiente:

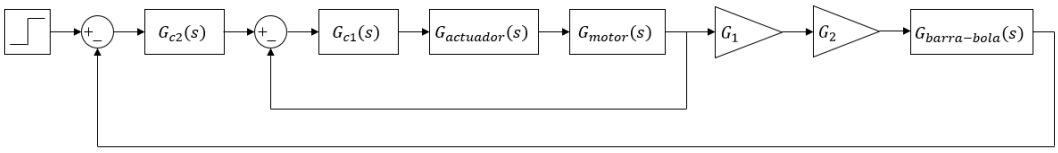

Figura A1-18. Lazo de control de la práctica 4

Se trata de un control en cascada, donde se tienen dos controladores:

- $G_{c1}(s)$: control esclavo. (Desarrollado en la práctica 2, se trata de un controlador PD para tiempo de establecimiento de 3 segundos)
- $G_{c2}(s)$: control maestro.

Para entender el lazo completo, este se explicará por partes:

- Lazo interno:

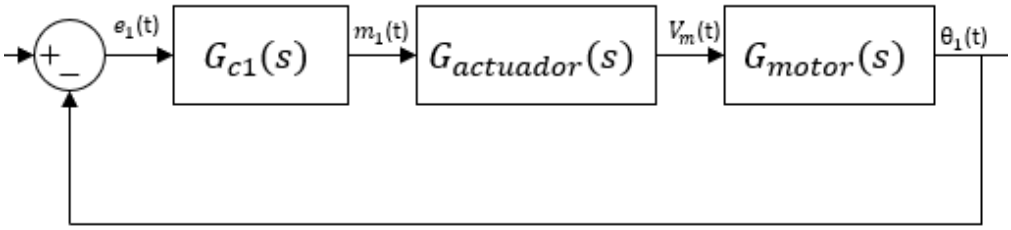

Figura A1-19. Lazo de control interno

El lazo interno es el mismo con el que se trabajó en la práctica 2, la señal $e_1(t)$, es el error, que será tratado por el control esclavo, generando así la variable manipulada $m_1(t)$. A través del actuador se obtiene $V_m(t)$ que se trata de la tensión de entrada del motor, y tras esto, se consigue la posición del motor en revoluciones $\theta_1(t)$.

Se debe recordar que:

$$G_{actuador}(s) = G_{PWM} * G_{Driver} = 0.12$$

$$G_{motor}(s) = \frac{0.2125}{s * (s + 11.9)}$$

$$G_{c1}(s) = 52.3 * (1 + 0.084 * s)$$

Por lo que se puede obtener $G_{LazoInterno}(s)$:

$$G_{LazoInterno}(s) = \frac{1.33}{s + 1.33}$$

- Lazo externo:

Volviendo al lazo externo y sustituyendo la equivalencia anteriormente descrita:

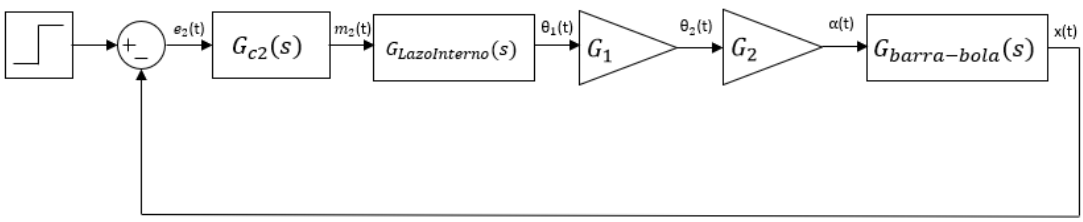

Figura A1-20. Lazo externo del sistema barra-bola

Como se puede ver, $e_2(t)$ se trata del error, producto de la diferencia entre la referencia de la posición de la bola y la posición de la bola medida.

Tras aplicar el algoritmo de control, se obtiene la señal de la variable manipulada $m_2(t)$, que en este caso actúa como referencia para el lazo de control esclavo.

La variable de proceso manipulada es $\theta_1(t)$ que como anteriormente se ha mencionado, se trata de la posición del motor en revoluciones. Sin embargo, a la entrada de $G_{barra\text{-}bola}(s)$ se necesita el ángulo de inclinación de la barra en radianes. Para ello se necesitan $G_1(s)$ y $G_2(s)$.

La función de la ganancia $G_1(s)$, es obtener la posición del motor en radianes $\theta_2(t)$.

$$G_1(s) = 2 * \pi$$

Se puede observar que ahora $G_{LazoInterno}(s)$ junto a G_1 cumplen la función de actuador para llevar a cabo el control del sistema barra bola.

La función de la ganancia $G_2(s)$, es obtener el ángulo de inclinación de la barra en radianes $\alpha(t)$ a partir de la posición del motor en radianes $\theta_2(t)$.

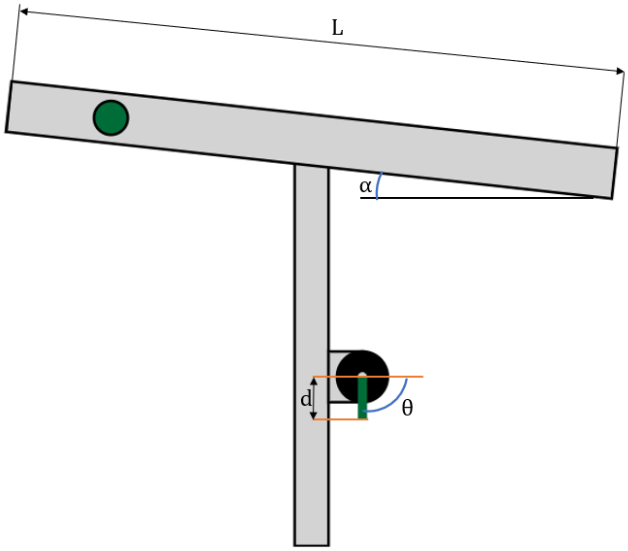

Figura A1-21. Captura del sistema barra-bola con variables necesarias para G_2

Siendo:

- d, offset del brazo en el motor, en el caso del laboratorio 3.5 centímetros.
- L, longitud de la barra, en el caso del laboratorio 50 centímetros.

$$G_2(s) = \frac{d}{L} = 0.07$$

La última dinámica es $G_{barra-bola}(s)$, la cual tiene como entrada α(t) y como salida la posición de la bola x(t), esta se trata de la misma que se obtuvo en la tercera práctica.

$$G_{barra-bola}(s) = \frac{7}{s * (s + 2.814)}$$

5.4. Uso de la interfaz.

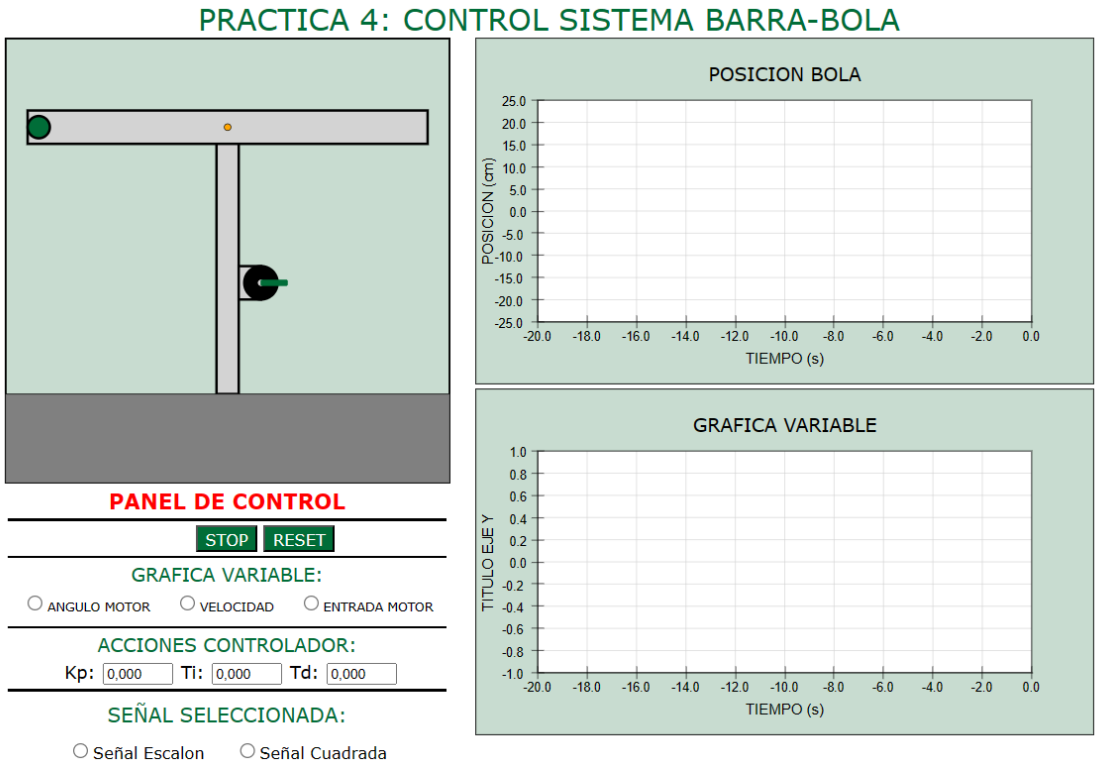

Figura A1-22. Interfaz gráfica Práctica 4

Lo primero que se debe de tener claro, es la expresión de la dinámica del controlador:

$$G_{control}(s) = k_p * (1 + T_d * s + \frac{T_i}{s})$$

A continuación, lo que se debe hacer, es una vez calculados los parámetros del controlador, ya sean los solicitados en las preguntas o los que el estudiante desee probar, introducirlos en el apartado de “ACCIONES CONTROLADOR” del panel de control de la interfaz.

Lo siguiente que se debe hacer es seleccionar que se desea visualizar en la gráfica variable, ofreciendo 3 posibilidades, el ángulo de inclinación del motor, la velocidad de la bola y la tensión de entrada que recibe el motor.

Tan pronto como se realice lo anteriormente descrito, al lado del botón “STOP” aparecerá otro botón “START”, si se pulsa, la simulación comenzará. En cualquier momento, esta se puede parar pulsando “STOP” y reanudarse de nuevo pulsando “START”, además se puede resetear la simulación pulsando “RESET”.

Dentro del panel de control, existe otra opción, que permite cambiar la señal de entrada (*set point*), teniendo dos opciones, una señal escalón y una señal cuadrada, esta opción se podrá cambiar en plena simulación. Si no se selecciona ninguna, de manera predeterminada, se usará una entrada señal escalón.

Es importante, una vez respondidas todas las preguntas, pulsar el botón de “ENVIAR”, para que la nota del estudiante quede grabada.

5.5. Conclusiones.

Tras realizar la práctica, el alumno debe de ser capaz de razonar qué tipo de controlador necesita según las especificaciones que se exijan, además de haberse familiarizado a trabajar con el lugar de las raíces.

El alumno, debe de comprender cómo afectan los parámetros del controlador a la respuesta, y puede llevar a cabo pruebas para ver cómo es la respuesta cuando el sistema es subamortiguado, críticamente amortiguado y sobreamortiguado.

5.6. Solución.

En la Figura A1-23, se pueden observar las respuestas correctas a las cuestiones que debe responder el alumno.

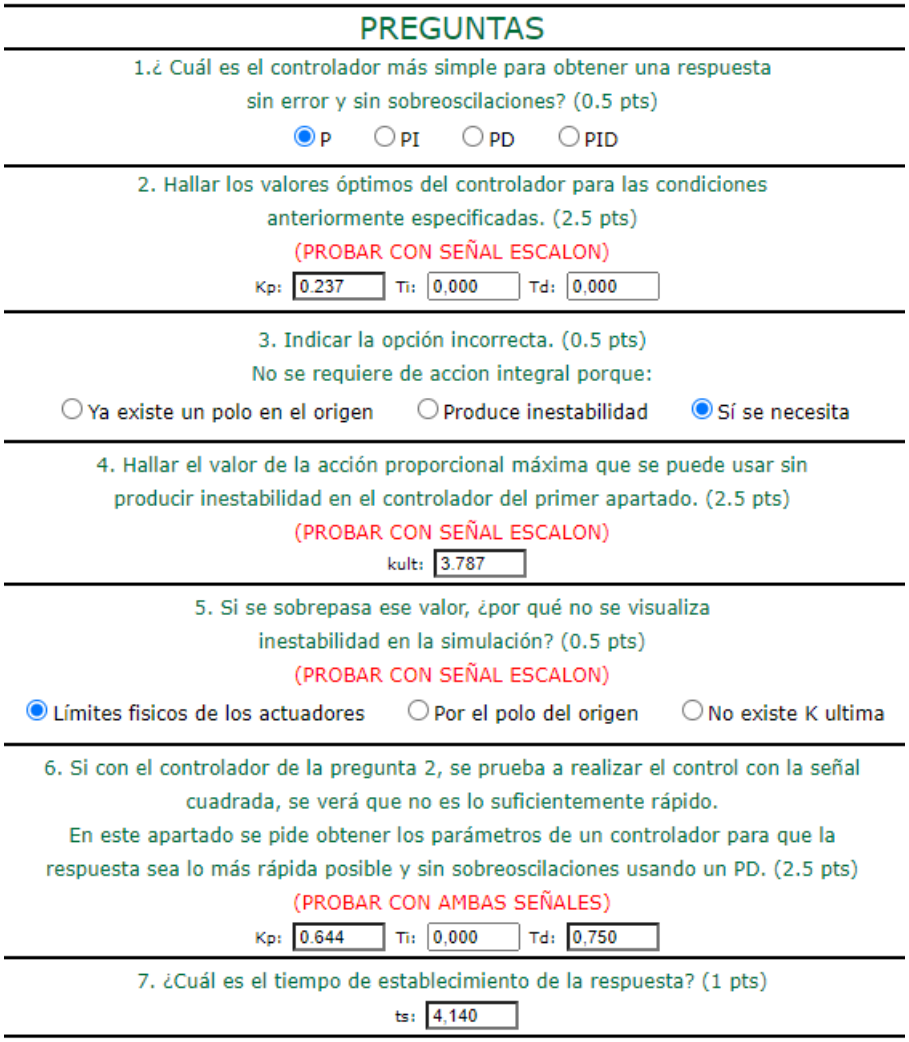

Figura A1-23. Soluciones de la práctica 4

6. Práctica 5: CONTROL DEL SISTEMA BARRA-BOLA. LAB REMOTO.

6.1. Introducción.

Esta práctica consiste en hallar los parámetros de un controlador PID, para llevar a cabo el control del sistema barra-bola, la principal diferencia con la práctica 4, es que, en este caso, se trata de un laboratorio remoto. De esta manera el estudiante podrá visualizar el funcionamiento del control sobre una estación real.

6.2. Objetivos.

El principal objetivo de esta práctica es que el alumno visualice el funcionamiento del control sobre el sistema barra-bola de manera remota, pudiendo interactuar así con él.

Para ello la práctica presenta la posibilidad de ingresar los parámetros del controlador que el alumno considere oportunos.

6.3. Manual de usuario.

Al acceder a la práctica el alumnado tendrá la interfaz que se ilustra en la Figura A1-24. Se observa que en: (a) la parte superior izquierda se presenta un vídeo en tiempo real del sistema barra-bola; (b) parte superior a la derecha una gráfica (aún vacía) de la posición de la bola en función del tiempo; (c) en la parte inferior izquierda hay un panel de control, donde de momento únicamente es visible un botón “PM” de puesta en marcha y tres campos numéricos para introducir los parámetros del controlador; y (d) en la parte inferior derecha, un panel del estado del control que únicamente indica que el control está en marcha.

Es importante destacar, que, en esta práctica, no se da la posibilidad de que el alumno encienda o apague el panel de iluminación, ya que, si se pudiese apagar o encender el panel mientras se realiza el control automático se producirían movimientos muy violentos por parte del motor al no detectar la posición de la bola, lo cual produciría problemas con la calibración de la cámara, debido a las vibraciones.

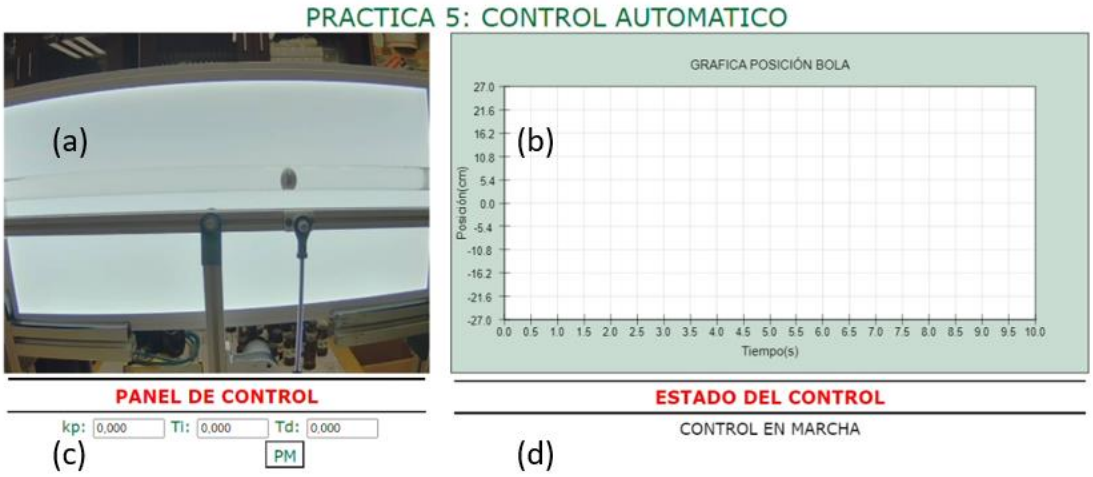

Figura A1-24. Interfaz de la práctica 5- control de la posición de un sistema barra-bola en un laboratorio remoto. (a)vídeo en tiempo real del sistema; (b) gráfica (aún vacía) de la posición de la bola en función del tiempo; (c) panel de control; (d) panel del estado del control.

Los pasos a seguir en la práctica son los siguientes:

1. Puesta en marcha: tras pulsar el botón “PM”, se verá cómo el motor gira, produciendo así el movimiento de la barra, la finalidad de llevar esto a cabo, es comenzar siempre desde un mismo punto de partida, en este caso, el extremo izquierdo. Una vez terminada la puesta en marcha se verá cómo el panel de iluminación parpadea, para indicar que esta ha finalizado. Además, se harán visibles dos botones:

    - **START:** permite comenzar el control del sistema barra-bola.
    - **STOP:** permite parar en cualquier momento el control del sistema barra-bola una vez este se haya iniciado.

    Ambos botones no serán habilitados hasta que el usuario no introduzca los parámetros del controlador.

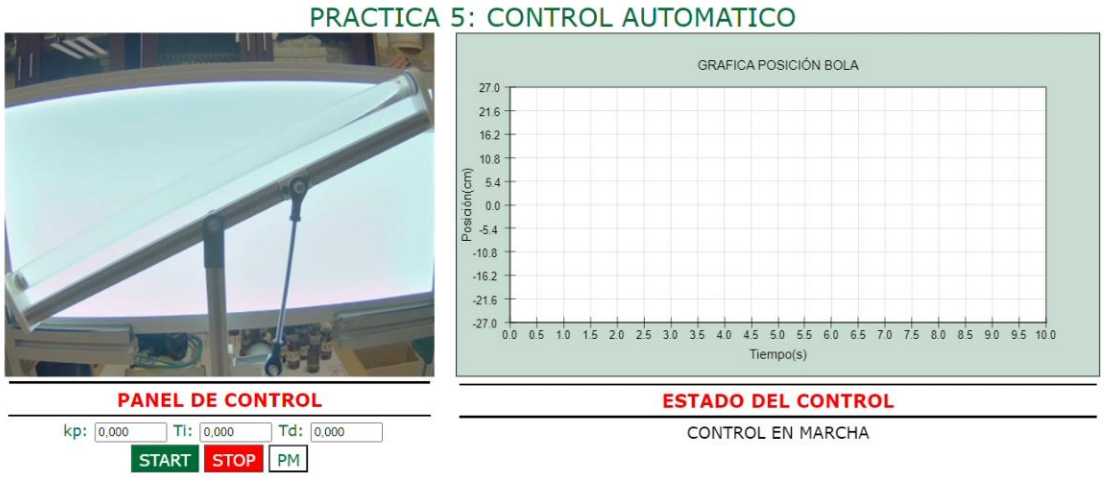

Figura A1-25. Interfaz de la práctica 5 durante la puesta en marcha

2. Inicio del control: el usuario ahora debe de introducir los parámetros del controlador, y tras esto, solo se debe de pulsar el botón “START”. A continuación, se podrá observar cómo, de manera automática, se lleva a cabo el control del sistema, mientras que, en la parte derecha de la interfaz, se grafica la posición de la bola.

   Si en algún momento del control, se observa que no funciona como debería, o que, con los parámetros introducidos, la bola no alcanza el centro de la bola, sólo se debe de pulsar “STOP”, y tras esto volver a pulsar “PM” para volver a la posición de inicio de la estación y volver a comenzar el control. Se debe de recordar, que el lazo de control es el siguiente:

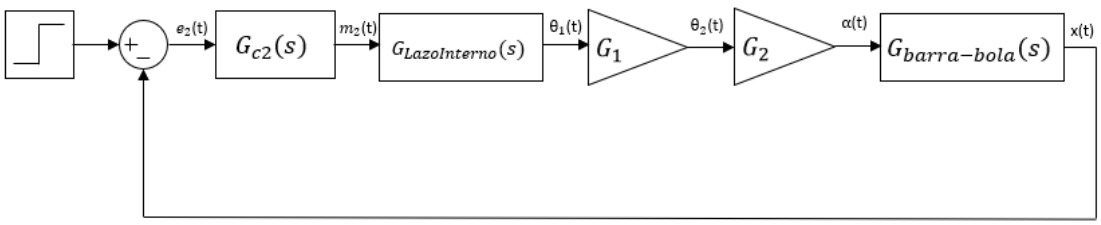

Figura A1-26. Lazo de Control del sistema barra-bola

Siendo:

$$G_{c2}(s) = k_p * \left(1 + T_d * s + \frac{T_i}{s}\right)$$

$$G_{LazoInterno}(s) = \frac{1.33}{s + 1.33}$$

$$G_1(s) = 2 * \pi$$

$$G_2(s) = 0.07$$

$$G_{barra-bola}(s) = \frac{7}{s * (s + 2.814)}$$

Además, para obtener la respuesta más óptima en el laboratorio remoto, se debe de elaborar un controlador PD, con el menor tiempo de establecimiento posible y sin sobreoscilaciones.

3. Fin del control: una vez llevada la bola al centro de la barra, el control habrá terminado, esto será indicado en el panel del estado de control.

   El usuario ahora tiene dos posibilidades, pulsar “PM” para reiniciar la estación y volver a llevar a cabo el control o pulsar “ENVIAR”, para que se grabe su puntuación en función de los parámetros del controlador introducidos. Tras esto, el usuario puede abandonar la práctica.

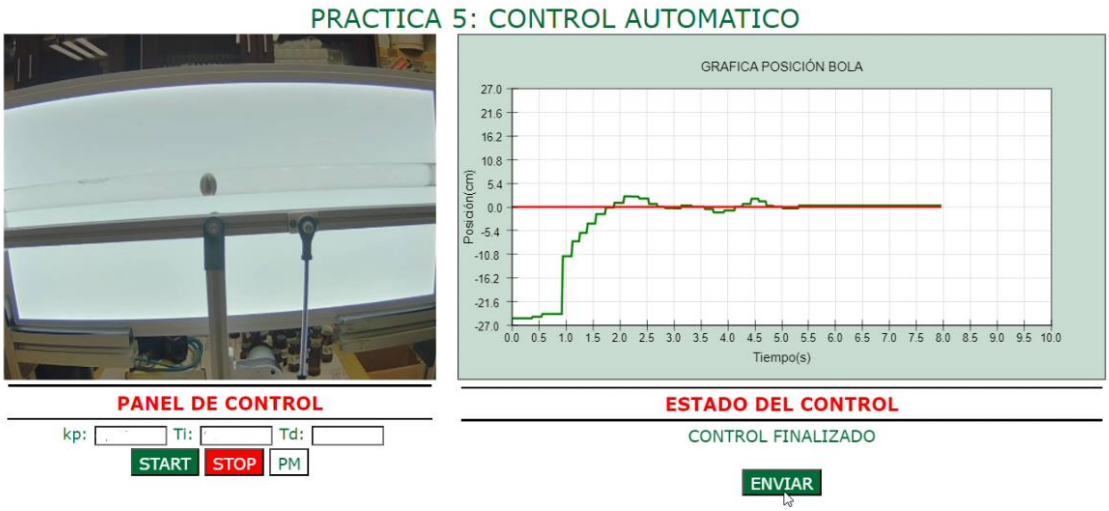

Figura A1-27. Interfaz gráfica de la práctica 5 al finalizar el control

6.4. Conclusiones.

El estudiante debe de terminar la práctica habiendo entendido cómo funciona el control del sistema barra-bola y que llevar a cabo un control automático ofrece resultados mejores que en el caso del control manual.

Además, la realización de esta práctica y todas las prácticas anteriores, debe haber permitido al estudiante aplicar los conceptos teóricos adquiridos a lo largo de la asignatura, en un caso real, en el cual se han seguido todos los pasos necesarios para desarrollar el control de un sistema complejo, como es un sistema barra-bola.

ANEXO 2: Manual de mantenimiento del laboratorio remoto.

En el siguiente anexo, se detallan los pasos que se deben seguir para llevar a cabo el mantenimiento del laboratorio remoto y así poder asegurar su correcto funcionamiento.

Lo primero que se debe entender, es que dicho mantenimiento se realizará de manera local y para ello, el primer paso es parar el servidor, debido a que este se encuentra activo en todo momento.

Para parar el servidor, se deben seguir las siguientes instrucciones:

1. Encontrar el identificador del proceso del servidor, para ello en una ventana de comandos de la Raspberry, se introducirá lo siguiente:

   ps aux | grep App

2. Una vez obtenido el identificador se dará la siguiente instrucción a la ventana de comandos:

   sudo kill (identificador)

Después de haber detenido el servidor, el siguiente paso será llevar a cabo la calibración de la cámara que se encarga de obtener la posición de la bola.

Esta calibración se realiza de manera local y manual, y para llevarla a cabo, se ha desarrollado un archivo.py, el cual se encuentra en la Raspberry Pi, y que permitirá visualizar la imagen que capta la cámara con unas referencias de posición, tal y como se puede observar en la Figura A2-1. Lo único que se debe de hacer, es manipular la posición de la cámara hasta conseguir que las referencias de posición coincidan con las esquinas inferiores del panel de iluminación y con el eje de giro de la barra, de esta manera la calibración sería completada.

Figura A2-1. Interfaz de calibración de la cámara Raspberry v2.1

El siguiente paso para continuar con el mantenimiento del laboratorio remoto, es asegurar que la configuración del servidor RIP, sea la correcta, para ello se recomienda acudir al capítulo 5.1.1, donde siguiendo las instrucciones correspondientes, se puede llevar a cabo la configuración en caso de algún cambio en las direcciones IP de los dispositivos involucrados en el laboratorio remoto.

Por otro lado, también se tendría que modificar la dirección IP del servidor en la configuración del cliente, tal y como se detalla en el capítulo 5.3.1, para así poder realizar de manera correcta la conexión entre servidor y cliente.

El siguiente paso del manual de mantenimiento, sería revisar las distintas conexiones entre los dispositivos, para ello se debe acudir al capítulo 3.2, para así poder comprobar que todas las conexiones entre elementos están realizadas correctamente.

Obviamente para comprobar las conexiones entre los distintos dispositivos, se debe cortar la alimentación de estos, para así evitar posibles accidentes.

El último paso, sería volver a poner en marcha el servidor, para ello bastaría con reiniciar la Raspberry, y el servidor comenzaría a funcionar automáticamente, debido al archivo generado y que se explicó en el capítulo 5.1.4.

De esta manera el mantenimiento estaría finalizado y la maqueta debería funcionar correctamente.